Synthetic Student Responses: LLM-Extracted Features for IRT Difficulty Parameter Estimation

Matías Hoyl[1]

[1]School of Education, Stanford University

Abstract

Educational assessment relies heavily on knowing question difficulty, traditionally determined through resource-intensive pre-testing with students. This creates significant barriers for both classroom teachers and assessment developers. We investigate whether Item Response Theory (IRT) difficulty parameters can be accurately estimated without student testing by modeling the response process and explore the relative contribution of different feature types to prediction accuracy. Our approach combines traditional linguistic features with pedagogical insights extracted using Large Language Models (LLMs), including solution step count, cognitive complexity, and potential misconceptions. We implement a two-stage process: first training a neural network to predict how students would respond to questions, then deriving difficulty parameters from these simulated response patterns. Using a dataset of over 250,000 student responses to mathematics questions, our model achieves a Pearson correlation of approximately 0.78 between predicted and actual difficulty parameters on completely unseen questions.

1 Introduction

Educational assessment plays a vital role in teaching and learning, but creating high-quality questions requires knowing their difficulty level: how challenging they are for students with different abilities. Currently, determining question difficulty requires extensive pre-testing with students followed by statistical analysis, an approach that is effective but creates significant delays and costs in developing new assessments.

For teachers developing classroom assessments, this process is often impractical, leading to tests with unknown psychometric properties. For large-scale assessment programs, the pre-testing requirement consumes substantial resources and limits the speed at which new content can be developed. Both scenarios highlight the need for methods that can predict question difficulty without requiring actual student testing.

This study addresses two specific research questions:

- How effectively can we estimate Item Response Theory difficulty parameters without student testing by modeling the student response process?
- To what extent do different feature types (LLM-extracted pedagogical features, linguistic characteristics, and semantic embeddings) contribute to the accuracy of difficulty predictions?

Rather than treating difficulty prediction as a direct regression problem, which is what most existing research does, we take an approach that mirrors how difficulty emerges in practice: through patterns of student responses.

Our method combines conventional question features with pedagogical insights extracted using Large Language Models (LLMs). These AI tools analyze questions to identify solution steps, required mathematical skills, cognitive complexity, and potential student misconceptions—aspects typically assessed by human experts. The LLM features function much like an experienced teacher would: reading the question and assessing its difficulty based on intuition and experience, combining multiple pedagogical signals to form a holistic judgment. By embedding this pedagogical knowledge in our predictions, we create a system that better reflects how questions function in educational settings.

The remainder of this paper is organized as follows. Section 2 reviews related work on difficulty prediction, highlighting existing approaches and gaps in the literature. Section 3 describes our dataset, which contains over 250,000 student responses to mathematics questions. Section 4 outlines our methodology, including feature engineering and our neural network architecture. Section 5 presents our experimental results, including the primary model's performance, an RMSE-based efficiency evaluation, and a feature ablation study. Finally, Section 6 discusses the implications of our findings, acknowledges limitations, and suggests directions for future research.

2 Related Work

2.1 Traditional Item Difficulty Estimation

Traditional approaches to difficulty estimation typically rely on either expert judgment or statistical analysis of student responses. Manual calibration by subject matter experts, while common, has been shown to be inconsistent and subjective [1, 2]. Items developed by experts often perform differently than expected when deployed in real settings.

Statistical calibration through pre-testing with actual students provides more reliable estimates but introduces significant delays before questions can be used operationally. This approach typically applies Item Response Theory (IRT) models to estimate difficulty parameters based on student response patterns [3]. While effective, this process is time-consuming and resource-intensive, creating a bottleneck in the development of high-quality assessments.

The limitations of these traditional methods have motivated research into automated approaches that can predict item difficulty from the question's textual content, potentially reducing or eliminating the need for extensive pre-testing.

2.2 Text-based Features for Difficulty Prediction

Research on text-based difficulty prediction has identified several categories of features that contribute to question complexity.

2.2.1 Linguistic Features

Linguistic characteristics of questions are strongly associated with their difficulty. Some studies have extracted lexical, syntactic, and semantic features to predict question complexity [4, 5]. Common linguistic features include:

- **Lexical features:** Word count, word frequency, word length, and vocabulary difficulty [6, 7]
- **Syntactic features:** Sentence length, syntactic complexity, and grammatical structures [5]
- **Semantic features:** Concept density, abstraction level, and semantic similarity between options [8]

Hickendorff [9] demonstrated that linguistic factors significantly impact student performance on mathematics assessments, even when controlling for mathematical complexity.

2.2.2 Domain-specific Features

Features specific to certain domains have been shown to improve prediction accuracy:

- **Mathematics:** Number of mathematical symbols, presence of graphs or figures, computation complexity [10]
- **Reading comprehension:** Text layout, presence of contextual cues, relationship between passages and questions [11]
- **Multiple-choice questions:** Similarity between options, plausibility of distractors [12, 13]

Toyama [14] emphasized that item difficulty is determined by the interplay of item characteristics, respondent characteristics, and the context in which assessment occurs.

2.3 Machine Learning Approaches for Difficulty Prediction

Early approaches to difficulty prediction relied on deterministic methods or simple regression models. Recent advances have introduced increasingly sophisticated machine learning techniques.

2.3.1 Classical Machine Learning Methods

Traditional machine learning algorithms have been widely applied to the challenge of difficulty prediction. Linear and logistic regression models have proven effective in establishing relationships between text features and difficulty levels, as demonstrated by Ha et al. [6] in their work on high-stakes medical exams. More sophisticated approaches include the R2DE model proposed by Benedetto et al. [10], which leverages random forests to simultaneously predict both IRT difficulty and discrimination parameters based on TF-IDF features extracted from question text. The literature also shows that feature-based ensemble methods have achieved notable success across various educational domains, with Ha et al. [6] demonstrating that combining different feature types can enhance prediction accuracy compared to single-feature approaches.

2.3.2 Neural Network Models

Deep learning approaches have increasingly been applied to difficulty prediction, with several notable innovations emerging in recent years. Recurrent Neural Networks, particularly those utilizing Long Short-Term Memory (LSTM) architectures, have demonstrated effectiveness in capturing sequential patterns within question text, as shown in the work of Huang et al. [15]. Building on this foundation, Qiu, Wu, and Fan [16] introduced a more sophisticated approach with their Document-enhanced Attention Network (DAN), which improves prediction accuracy by enriching questions with relevant domain documents. This contextual enhancement allows the model to better understand the relationship between questions and their subject matter. Further advancing the field, Xue et al. [17] explored transfer learning techniques, demonstrating that models pre-trained on related tasks such as response time prediction can be effectively fine-tuned for difficulty estimation, leveraging knowledge gained from one educational assessment task to improve performance on another.

2.3.3 Transformer-based Approaches

The most recent research has explored transformer-based models for difficulty prediction, leveraging their superior ability to capture contextual relationships. Benedetto et al. [18] compared BERT and DistilBERT models for difficulty estimation, finding that transformer models outperform previous approaches by up to 6.5% in terms of RMSE. Building on this work, Gombert et al. [19] employed scalar-mixed transformer encoders with specialized regression heads, showing significant improvements over baseline models. Further advancing the field, Kapoor et al. [20] incorporated embeddings from various LLMs (ModernBERT, BERT, and LLAMA) alongside linguistic features, achieving a correlation of 0.77 between true and predicted difficulty.

A key advantage of transformer models is their ability to capture complex semantic relationships within text without requiring extensive feature engineering. Aradelli [21] demonstrated that fine-tuning pre-trained transformers on domain-specific corpora further enhances prediction accuracy. These approaches represent the current state-of-the-art in difficulty prediction, combining the contextual understanding capabilities of transformer architectures with specialized training techniques to achieve unprecedented levels of accuracy in estimating question difficulty from text alone.

2.4 Feature Extraction Using Language Models

Recent studies have explored the use of Large Language Models (LLMs) not just for prediction but also for feature extraction:

- **Procedural complexity:** Using LLMs to quantify the number of steps required to solve problems Liu et al. [22]
- **Skill identification:** Extracting the specific skills required to answer questions Didolkar et al. [23]
- **Cognitive level assessment:** Classifying questions according to Bloom's taxonomy Scaria, Dharani Chenna, and Subramani [24]
- **Misconception analysis:** Identifying potential student misconceptions associated with questions Sadihin, Rodriguez, and Chen [25]

These approaches leverage the reasoning capabilities of LLMs to extract pedagogically meaningful features that might be difficult to capture through conventional feature engineering approaches.

2.5 Gaps in Current Research

Despite significant advances in difficulty prediction methods, two important gaps remain in current research.

First, while transformer-based models have shown remarkable results, often outperforming traditional machine learning approaches by leveraging contextual relationships within question text, there has been limited exploration of using Large Language Models (LLMs) to extract pedagogically meaningful features from questions. Most studies rely on conventional linguistic features or embedding-based approaches, neglecting the potential of LLMs to identify more complex attributes, such as procedural complexity, specific mathematical skills, cognitive levels, and potential misconceptions.

These LLM-extracted features could provide richer signals about question characteristics that are difficult to capture through traditional feature engineering approaches. The few studies that have begun exploring LLM-based feature extraction have shown promising results, but this approach remains underutilized in the difficulty prediction literature.

Second, most current research approaches difficulty prediction as a direct regression problem, where models are trained to predict difficulty parameters directly from question text and features. This direct approach, while intuitive, bypasses a crucial intermediate step: modeling how students with varying ability levels would actually respond to these questions.

Few studies have explored the alternative approach of first simulating student responses to questions and then deriving difficulty parameters from these simulated response patterns. This simulation-based approach offers several advantages over direct prediction. It better mimics the actual process through which question difficulty is determined in real educational settings, allowing difficulty parameters to emerge naturally from student performance patterns rather than being directly predicted. Additionally, it provides a more interpretable model that captures both question characteristics and student abilities, while also producing additional useful metrics beyond difficulty, such as discrimination parameters and expected response patterns.

This gap presents an opportunity to develop more robust difficulty estimation methods. If a model can successfully simulate how students would respond to new questions, it could potentially be applied to any unseen question to estimate difficulty parameters without requiring actual student testing. Such an approach would significantly reduce the resource burden of developing high-quality assessments while maintaining psychometric validity.

The present study addresses both of these gaps by (1) leveraging LLMs to extract complex pedagogical features from questions and (2) developing a neural network model that first predicts student responses and then derives IRT difficulty parameters from these predictions, offering a more comprehensive approach to the difficulty estimation problem.

3 Data

The dataset for this study comes from Zapien, an adaptive math learning platform in Chile. It includes 251,851 student answers to 4,696 unique math questions, provided by 1,875 students.

Since the original question text and answer options were in Spanish, we first translated them to English using Google's Gemini 2.0 Flash model that excels at translation. This step was crucial for optimizing the performance of the English-based text embedding models used later in our feature engineering.

Next, we established benchmark difficulty values for each question. Using the entire dataset of 251,851 student responses, we applied a standard 1-Parameter Logistic (1PL) Item Response Theory model. This model estimates a difficulty parameter for each question based on the observed patterns of correct and incorrect answers across all students. These benchmark parameters represent the "ground truth" difficulty derived from real student performance.

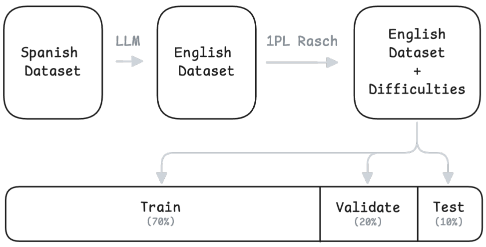

Figure 1: *Data processing pipeline.*

Finally, to train and evaluate our prediction model, we split the 4,696 questions into three sets: training (70%), validation (20%), and a holdout test set (10%). This split was done using stratified sampling based on both the difficulty calculated earlier and the average student correctness for each question. Stratification ensures that all three sets have comparable distributions of question difficulty and student performance patterns.

Split	Questions	Answers
Training	3,286 (70%)	178,031 (71%)
Validation	940 (20%)	49,100 (19%)
Holdout Test	470 (10%)	24,720 (10%)

Table 1: *Dataset splits.*

4 Methods

This section outlines our methodological approach. Our method predicts question difficulty in a two-stage process. First, we train a neural network to predict whether a student will answer a given mathematics question correctly. This network uses three main types of information: (1) student representation (as a vector of features that the model learns from the training data), (2) a rich set of question features (including linguistic characteristics, structural properties, and pedagogical insights extracted by LLMs), and (3) semantic embeddings of the question text itself. Second, we use the network's correctness predictions across many simulated student-question interactions to derive Item Response Theory (IRT) difficulty parameters for each question.

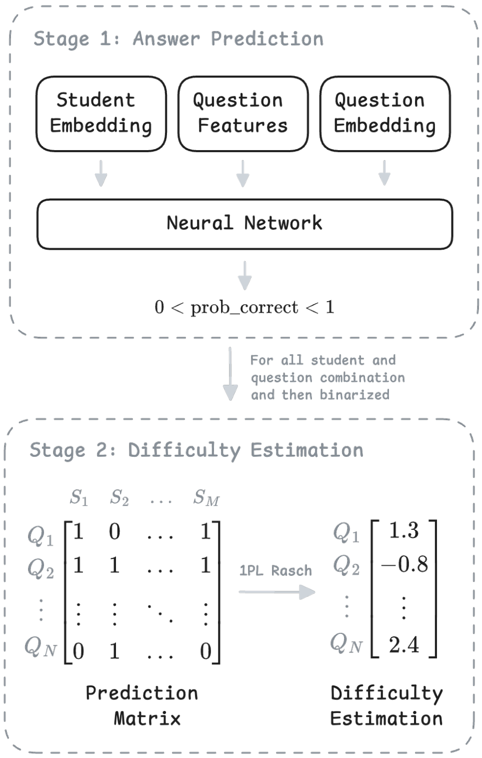

Figure 2: *Full prediction and estimation pipeline. The neural network makes a prediction for each student-question interaction, then we use the 1PL IRT model to estimate the difficulty parameters using a prediction matrix.*

4.1 Stage 1: Neural Network for Correctness Prediction

Our neural network model is designed to predict the probability of a student answering a question correctly. It takes three primary types of inputs:

1. **Question Embedding**: A 768-dimensional embedding vector representing the formatted question text and answer options.
2. **Question Features**: A vector of 48 engineered features capturing question characteristics (linguistic, structural, and LLM-derived pedagogical aspects, including one-hot encoded categorical features).
3. **User Embedding**: A learned embedding vector representing the student, for capturing student-question interactions.

4.1.1 Question Text Embeddings

To capture the semantic meaning of the entire question context, we generated contextual embeddings for the question title combined with its answer options. The text was formatted as:

```
Question:  [problem_statement]
Correct Answer:  [correct_option]
Wrong Answer 1:  [wrong_option_1]
...
Wrong Answer 4:  [wrong_option_4]
```

This formatted text was processed using the `nomic-ai/modernbert-embed-base` model to generate 768-dimensional embeddings, which were L2 normalized. These serve as a direct input to the neural network.

4.1.2 Linguistic and Structural Features

A set of numerical features was engineered to capture textual complexity, mathematical content, and structural characteristics of the questions and their answer options. These features include:

- **From Question Title/Text:** `question_word_count`, `question_char_count`, `question_avg_word_length`, `question_digit_count`, `question_special_char_count`, `question_mathematical_symbols`, `question_latex_expressions`.
- **From Answer Options:** `jaccard_similarity_std` (standard deviation of Jaccard similarities between option pairs), `avg_option_length`, `avg_option_word_count`, `Answer_Length_Variance`, `Has_NoneAll_Option` (binary, treated as numerical), `Option_Length_Outlier_Flag` (binary, treated as numerical).
- **Structural/Metadata:** `has_image` (binary, treated as numerical).

These features are part of the `BASE_NUMERICAL_FEATURE_COLS` used in our primary model script.

4.1.3 Options Embeddings and Derived Features

While individual option embeddings were also generated using `nomic-ai/modernbert-embed-base` for each answer choice (A through E), these raw embeddings were not directly fed into separate pathways in the final neural network. Instead, specific numerical features were pre-calculated from these option embeddings and included in the `BASE_NUMERICAL_FEATURE_COLS`. These derived features are:

- **Correct_Distractor_CosineSim_Mean:** The mean cosine similarity between the embedding of the correct option and the embeddings of each incorrect (distractor) option.
- **Distractor_Embedding_Distance_Mean:** The mean Euclidean distance between the embeddings of all pairs of distractor options.

These metrics provide insights into the semantic similarity between the correct answer and distractors, and among the distractors themselves.

4.1.4 Feature Extraction Using LLMs

We utilized Google's Gemini models (primarily `gemini-2.0-flash` for simpler feature extractions and `gemini-2.5-flash` or `gemini-2.5-pro` for more complex ones like archetype classification, as detailed in Appendix **??**) to extract pedagogically relevant features from the question text and, where applicable, provided solutions. To ensure stability and mitigate the probabilistic nature of LLMs, features were typically generated over N runs (e.g., 3 runs for numerical estimates like information gap, or 1 run for simpler classifications) and then aggregated (e.g., by averaging numerical values or using the mode for categorical ones). The prompts used aimed to elicit chain-of-thought reasoning from the LLM before providing the final structured JSON output. It is important to note that while a broader range of pedagogical features (such as a detailed taxonomy of mathematical skills) were explored during development, the final reported model incorporates the LLM-derived features listed below.

Procedural Complexity and Cognitive Demand

- **avg_steps**: The average number of discrete, pedagogically atomic solution steps required to

solve the question, determined by LLM analysis of the solution process (averaged over 3 runs). The prompt guided the LLM to break down solutions in a manner appropriate for student instruction (see Appendix **??** for a conceptual prompt example, specific prompt for this automated feature might differ but follows this intent).
- **level**: The primary cognitive level assessed by the question, based on Bloom's Taxonomy (1-6, see Appendix **??**), determined by LLM analysis (majority vote over 3 runs). This is an ordinal feature treated as numerical. (See Appendix **??** for the conceptual prompt framework).
- **num_misconceptions**: The average number of potential student misconceptions associated with the question, identified by the LLM (averaged over 3 runs). The prompt asked for an exhaustive, atomic list (see Appendix **??** for a conceptual prompt framework).

Mathematical and Symbolic Content Features (LLM-derived)

- **Mathematical_Notation_Density**: An LLM-derived measure assessing the density or complexity of mathematical notation. (While related to nesting depth, specific extraction might involve a nuanced prompt focusing on overall symbolic density, see Appendix **??** for the related nesting depth prompt which captures aspects of this).
- **Max_Expression_Nesting_Depth**: The maximum nesting depth of mathematical expressions (e.g., parentheses, functions, fractions) found in the question, determined by LLM analysis (mode over 3 runs, see Appendix **??**).
- **Ratio_Abstract_Concrete_Symbols**: The ratio of abstract symbols (variables) to concrete numerical values in the question, calculated from LLM counts (see Appendix **??**).
- **Has_Abstract_Symbols**: Binary indicator (1/0) if abstract symbols (beyond common variables) are present, determined by LLM analysis (1 run, see Appendix **??**).
- **Units_Check**: Binary indicator (1/0) if checking or converting units is significantly involved, determined by LLM analysis (1 run, see Appendix **??**).

Question-Answer Relationship and Context Features (LLM-derived)

- **Question_Answer_Info_Gap**: An LLM-estimated measure (1-4 scale) of the information gap between the question stem and the knowledge needed to derive the answer (averaged over 3 runs, see Appendix **??**).
- **RealWorld_Context_Flag**: Binary indicator (1/0) if the question is set in a real-world context versus purely abstract, determined by LLM analysis (1 run, see Appendix **??**).

Distractor Plausibility Features (LLM-derived)

- **LLM_Distractor_Plausibility_Max**: The average (over 3 runs) of the maximum plausibility scores (1-5 scale) assigned by an LLM among the distractor options for a given question (see Appendix **??**).
- **LLM_Distractor_Plausibility_Mean**: The average (over 3 runs) of the mean plausibility scores (1-5 scale) assigned by an LLM to the distractor options for a given question (see Appendix **??**).
- **Extreme_Wording_Option_Count**: Count of options containing extreme wording (e.g., "always", "never"). (This feature might be rule-based or LLM-derived; if LLM-derived, a specific prompt would assess this, otherwise it's pattern matching.)

Categorical Pedagogical Features (LLM-derived, One-Hot Encoded) The following LLM-derived features were treated as categorical and subsequently one-hot encoded before being fed into the neural network:

- **Knowledge_Dimension**: The primary dimension of knowledge assessed (Factual, Conceptual, or Procedural), determined by LLM analysis (1 run, see Appendix **??**).
- **Most_Complex_Number_Type**: The highest level of numerical complexity (1-5 scale: Integer to Complex Number) present in the question or options, determined by LLM analysis (1 run, see Appendix **??**).
- **Problem_Archetype**: The general type or structure of the problem (e.g., Word Problem - Calculation, Equation Solving), determined by LLM analysis using `gemini-2.5-pro` (1 run, see Appendix **??**).

The combination of the raw numerical features listed above (linguistic, structural, LLM-derived numerical/binary) and the one-hot encoded versions of these three categorical features results in the 48 final numerical input features for the neural network.

4.1.5 User Embeddings

The last piece of the puzzle in our modeling approach was incorporating user information, since we had student answer data for each question. Accounting for individual student abilities was essential for accurate predictions, but presented significant technical challenges. We initially attempted to incorporate user IDs as one-hot encoded features in a LightGBM model, but this approach proved problematic. With 1,867 binary features for users compared to only 146 question-related features, the model essentially functioned as a user classifier rather than a question difficulty estimator. This structural imbalance resulted in predictions with minimal variation across questions for the same user (standard deviation only 0.026), compromising our ability to assess question characteristics.

To address this limitation, we implemented a neural network with a dedicated user embedding layer that mapped each user ID to an 8-dimensional vector space. This approach reduced the feature dimensionality while creating dense, continuous representations of user abilities. The embedding layer contained a learnable matrix of dimensions $(n\_users \times embedding\_dim)$ that captured latent ability factors conventional one-hot encoding could not efficiently represent. This embedding-based approach successfully balanced user and question influences, enabling our model to make meaningfully different predictions for different questions presented to the same user, while ensuring user characteristics did not overshadow the question features we were primarily interested in analyzing.

4.2 Modeling Approach

Our neural network model is designed to predict the probability of a student answering a question correctly. It takes three primary types of inputs:

1. **User Input**: An embedding vector representing the student.
2. **Numerical Input**: A vector of 48 engineered features capturing question characteristics (linguistic, structural, and LLM-derived pedagogical aspects, including OHE categorical features).
3. **Text Embedding Input**: A 768-dimensional embedding vector representing the formatted question text.

The architecture (as detailed in `modules/neural_net.py` and configured by `run_best_model_no_filter.py`) is as follows:

- The user ID is passed through an embedding layer (8-dimensional output), followed by flattening and a dropout layer (rate 0.25).
- The 48 numerical features are processed by a dense layer (32 units, ReLU activation, L2 regularization of 0.0005), followed by a dropout layer (rate 0.25).
- The 768-dimensional question text embedding is processed by a dense layer (32 units, ReLU activation, L2 regularization of 0.0005), followed by a dropout layer (rate 0.25).
- The outputs of these three pathways are concatenated.
- This concatenated vector is then passed through two shared dense layers (64 units and then 32 units, both with ReLU activation, L2 regularization of 0.0005, and dropout rate of 0.25 after each).
- The final layer is a single dense unit with a sigmoid activation function to output the probability of a correct answer.

Component/Hyperparameter	**Value/Setting**
User Embedding Dimension	8
Dense Layer Units (Input Pathways)	32 (for numerical feat
Dense Layer Units (Shared)	64, then 32
Activation Function (Hidden Layers)	ReLU
L2 Regularization (All weight matrices)	0.0005
Dropout Rate (All dropout layers)	0.25
Optimization	Adam optimizer
Learning Rate	0.0002 (2e-4)
Loss Function	Binary cross-entropy
Metrics	Accuracy, AUC
Batch Size	1024
Maximum Epochs	60
Early Stopping Patience	12 (monitors `val_los`
Restore Best Weights (Early Stopping)	True

Table 2: *Neural network architecture and training hyperparameters for the best performing model (`config_F_lower_lr_more_patience`).*

4.3 Prediction Task

After training the neural network on 70% of the questions and validating on 20%, we used the model to generate predictions for the 10% completely unseen questions in the holdout test set. For each of the 470 holdout questions, we generated predictions for all unique users present in the training/validation data (approximately 1,862 users were involved in the final prediction matrix for the test questions as per the ablation script logs). We first preserved the raw probability outputs from the model, representing each student's likelihood of answering correctly. We then applied a 0.5 threshold to these probabilities to convert them to binary predictions (0 for incorrect, 1 for correct). This conversion produced a correctness matrix where each cell contained either 0 or 1.

We then used these binarized correctness matrices as inputs to an Item Response Theory estimation process. We implemented a 1-Parameter Logistic (1PL) IRT model to estimate question difficulty parameters. The parameter estimation employed a maximum likelihood approach using TensorFlow.

5 Results

5.1 Model Performance

Our neural network model achieved strong performance in predicting student correctness. The following metrics were obtained on the holdout test set (470 questions) using the best performing model configuration (`config_F_lower_lr_more_patience`):

Metric	Value
Accuracy	0.7529
AUC-ROC	0.7789
Precision	0.8010
Recall	0.8719
F1 Score	0.8350

Table 3: *Classification performance metrics on the holdout test set.*

These metrics indicate the model's robust ability to predict student responses on unseen questions, providing a solid foundation for the subsequent IRT parameter estimation from these predictions.

5.2 IRT Parameter Estimation

Our primary objective was to estimate 1-Parameter Logistic (1PL) IRT difficulty parameters from the neural network's binarized predictions on the holdout test set. The alignment between these NN-derived difficulty parameters and the benchmark 1PL IRT difficulty parameters (calculated from the full original student response dataset) is shown below for our best performing full-feature model (equivalent to Model 4 in the ablation study):

Metric	Value
Pearson correlation	0.7791
Spearman rank correlation	0.7606
Number of common questions	470

Table 4: *1PL IRT difficulty parameter estimation metrics on the holdout test set for the full feature model.*

The high correlation values demonstrate a strong positive relationship between the difficulty parameters estimated via our modeling approach and those derived from actual student response data.

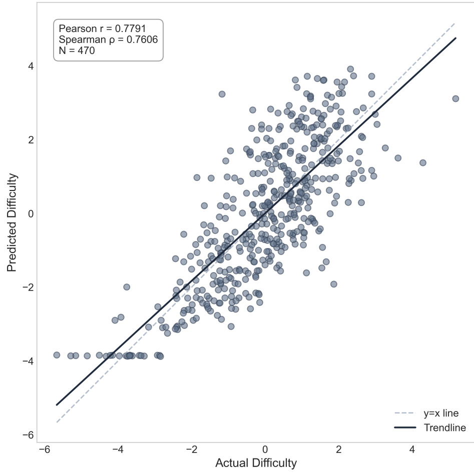

Figure 3: *Correlation between benchmark 1PL IRT difficulty and NN-derived 1PL difficulty (Model 4 - Full Features) on the holdout test set.*

As shown in Figures **??** and **??**, our model captures both the relative ordering and, to a good extent, the distribution of question difficulties.

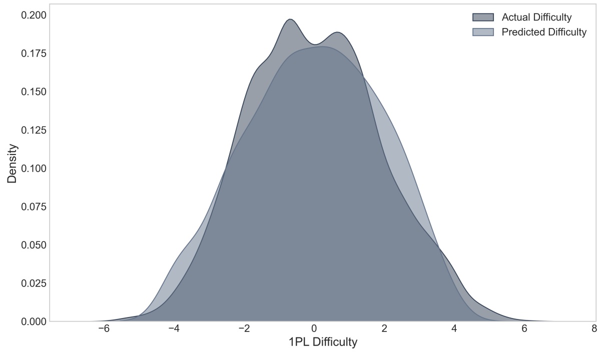

Figure 4: *Distribution comparison of benchmark 1PL IRT difficulty and NN-derived 1PL difficulty (Model 4 - Full Features) for the holdout test set.*

5.3 RMSE-Based Efficiency Evaluation

To further contextualize the performance of our NN-derived 1PL difficulty parameters, we conducted an RMSE-based efficiency evaluation. This analysis compares the accuracy of our model's difficulty estimates to those obtainable from traditional 1PL IRT estimations using varying amounts of real student data for the same 470 holdout questions. The ground truth for this comparison was established by calculating 1PL IRT difficulty parameters using all available answers for these 470 questions from the original unfiltered dataset.

After aligning the scale (mean and standard deviation) of our NN-derived 1PL difficulties to these ground truth difficulties, the **NN-Derived 1PL Model RMSE was 1.1857**.

Figure **??** illustrates this comparison, showing the

RMSE achieved by traditional 1PL IRT with different sample sizes against our model's performance. Table **??** provides a subset of these data points.

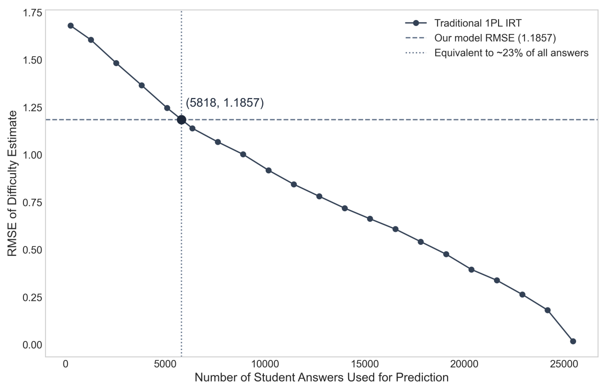

Figure 5: *RMSE-Based Efficiency of NN-Derived Difficulty Estimates compared to Traditional 1PL IRT using varying amounts of real student data.*

% Real Data Used	Approx. Answer Count	Simulated RMSE
1.0%	254	~1.6807
5.0%	1,272	~1.6059
10.0%	2,544	~1.4834
15.0%	3,816	~1.3666
20.0%	5,088	~1.2475
Our Model's Performance	**(NN-derived, 5,818 equiv.)**	**1.1857**
25.0%	6,360	~1.1399
30.0%	7,632	~1.0680
50.0%	12,720	~0.7815
100.0%	25,440	~0.0185

Table 5: *RMSE of 1PL IRT difficulty estimates using varying amounts of real student data for 470 holdout questions, compared to NN-derived difficulty RMSE.*

This evaluation indicates that our NN-derived 1PL difficulty estimates achieve an accuracy (RMSE of 1.1857) comparable to using approximately **5,818 real student answers** in a traditional 1PL IRT estimation for these same 470 holdout questions. This represents approximately **22.87%** of the total available real responses for these items in the holdout set. This suggests a significant potential for data efficiency using our modeling approach.

5.4 Feature Ablation Study

To investigate the contribution of different feature types to the model's performance in predicting student correctness and subsequent IRT difficulty estimation, a feature ablation study was conducted. This study employed the same 1PL vs. 1PL comparison framework and neural network hyperparameters as our main reported model.

5.4.1 Feature Buckets for Ablation Study

Features were incrementally added to a baseline of text embeddings:

1. **Model 1 (Embeddings Only)**: Utilized only the `nomic-ai/modernbert-embed-base` text embeddings of the formatted question text.
2. **Model 2 (Embeddings + Question Features)**: Added linguistic and structural features derived from the question title and text (e.g., `question_word_count`, `has_image`, `Has_Abstract_Symbols`). This corresponds to Bucket 1 in `modeling_strategy.md`.
3. **Model 3 (Embeddings + Question + Option Features)**: Added linguistic and structural features derived from the answer options (e.g., `jaccard_similarity_std`, `avg_option_length`, `Has_NoneAll_Option`). This corresponds to Bucket 1 + Bucket 2 in `modeling_strategy.md`.
4. **Model 4 (Embeddings + Question + Option + LLM Features)**: Added LLM-derived pedagogical features, both numerical (e.g., `avg_steps`, `level`, `num_misconceptions`, `Mathematical_Notation_Density`) and categorical (OHE: `Knowledge_Dimension`, `Most_Complex_Number_Type`, `Problem_Archetype`). This corresponds to Bucket 1 + Bucket 2 + Bucket 3 in `modeling_strategy.md` and represents the full feature set similar to our main reported model.

5.4.2 Ablation Results

Table **??** summarizes the performance of these four models on the test set.

Model Configuration	Test AUC	Test Acc.	Test F1	Pearson (1PL)	Spearman (1PL)
Model 1: Embeddings Only	0.7741	0.7518	0.8359	0.6822	0.7064
Model 2: Emb. + Question Feat.	0.7743	0.7524	0.8351	0.6842	0.7065
Model 3: Emb. + Q & Opt. Feat.	0.7753	0.7513	0.8343	0.6758	0.7044
Model 4: Full Model (Emb. + Q & Opt. + LLM)	0.7818	0.7530	0.8351	0.7791	0.7606

Table 6: *Feature ablation study results: correctness and 1PL vs. 1PL difficulty correlation metrics.*

5.4.3 Ablation Study Conclusions

This ablation study reveals the following insights:

- **Baseline (Model 1: Embeddings Only):** Text embeddings alone achieve a Pearson correlation of ~0.682 and Spearman of ~0.706 for 1PL difficulty. Correctness AUC is ~0.774.
- **Impact of Non-LLM Question and Option Features (Model 2 & 3 vs. Model 1):** Adding traditional linguistic/structural question features (Model 2) maintained similar performance. Further adding option features (Model 3, which now includes embedding-derived option metrics) showed a slight dip in Pearson correlation

(~0.676) but maintained Spearman correlation and AUC, suggesting these foundational features provide a stable base but don't dramatically lift performance over embeddings alone for difficulty correlation in this setup.

- **Impact of LLM-Derived Features (Model 4 vs. Model 3):** The most significant improvement came from adding the LLM-derived features. Model 4 (full model) showed a substantial increase in difficulty correlation (Pearson ~0.779, Spearman ~0.761) compared to Model 3. Correctness AUC also increased to ~0.782. This highlights the strong contribution of the LLM-generated features for capturing nuances related to question difficulty.

These results underscore the significant contribution of the LLM-extracted features for capturing aspects of question difficulty that are not fully encompassed by text embeddings or basic structural/linguistic features alone. The full feature set in Model 4 demonstrates the strongest performance, aligning closely with our main reported model's capabilities.

6 Discussion

6.1 Conclusions

Our research demonstrates that Item Response Theory difficulty parameters can be accurately estimated without traditional student testing by combining linguistic features with pedagogical insights extracted using Large Language Models. The final model achieved a Pearson correlation of approximately 0.78 (Spearman correlation ~0.76) between its predicted 1PL IRT difficulty parameters and benchmark 1PL parameters derived from actual student data, when evaluated on completely unseen questions. This approach addresses a significant challenge in educational assessment: the need for efficient methods to determine question difficulty that don't rely on resource-intensive pre-testing.

The strong performance of our neural network model in predicting student correctness (test set F1 score of 0.8350, AUC of 0.7789) confirms that our feature engineering approach effectively captures many factors influencing student performance. By integrating traditional linguistic and structural features, semantic embeddings of question text, and LLM-extracted pedagogical insights (such as solution step count, cognitive level, and misconception count), we created a rich representation of question characteristics.

Most importantly, the RMSE-based efficiency evaluation showed that our NN-derived 1PL difficulty estimates (RMSE of 1.1857 after scale alignment) are comparable in accuracy to using approximately 5,800 real student answers (representing about 22.87% of the available data for the holdout questions) in a traditional 1PL IRT estimation. This highlights a substantial potential for data efficiency and cost reduction in assessment development.

Furthermore, the feature ablation study confirmed the significant contribution of LLM-derived pedagogical features. When added to a baseline of text embeddings and other non-LLM structural/lexical features, the LLM features provided a notable boost in the correlation between predicted and benchmark IRT difficulty parameters. This supports the claim that these advanced, AI-extracted features are valuable for capturing nuances of question difficulty.

These findings suggest that our simulation-based approach can reliably estimate question difficulty, potentially accelerating assessment development while maintaining psychometric quality, and opens new avenues for AI-assisted educational assessment design.

6.2 Limitations

Despite the promising results, several limitations of our study should be acknowledged:

Generalizability : Our dataset originates from Chile, which may limit the generalizability of our findings to student populations elsewhere. Educational systems, curricula, and pedagogical approaches vary across countries and cultures, potentially affecting how question features relate to difficulty in different contexts.

LLM Stability : Large Language Models are inherently probabilistic. Even though we attempted to ensure stability by making multiple LLM calls and aggregating results for features like 'avg_steps' and 'num_misconceptions', each new call could potentially yield different outputs. This variability introduces some uncertainty into the LLM-extracted features, which could affect the consistency of our predictions.

Model Interpretability : The neural network we employed is essentially a black box that performs well on the prediction task but offers limited insight into the relative importance of different features beyond what was shown in the ablation study. This makes it challenging to draw definitive conclusions about which specific question characteristics contribute most significantly to difficulty within the combined model.

Model Sensitivity : Additionally, our results may vary depending on which LLM vendor we use (e.g., OpenAI, Anthropic, Google) and which inference parameters we select (e.g., temperature, top-p sampling). These choices can significantly impact the quality and consistency of the LLM-extracted features that feed into our model.

6.3 Future Work

In future iterations of this research, we plan to address these limitations through several targeted improvements:

Dataset Expansion : We are currently in discussions to obtain the ASSISTments dataset to validate our approach on a different student population. This additional dataset would allow us to test the generalizability of our method across different educational contexts and student demographics.

Baseline Comparisons and Feature Analysis : The feature ablation study provided initial insights. Further systematic removal or addition of specific LLM-derived features or other feature groups can help to more precisely quantify their individual and combined contributions to prediction accuracy and difficulty alignment. Comparing against simpler regression models using only subsets of features would also be beneficial.

Methodological Variations : We will test variations of the model architecture, different LLM providers or versions for feature extraction, and alternative embedding models to evaluate the sensitivity of our results to these methodological choices. This exploration will help identify the most robust and effective configuration for difficulty prediction.

Refinement of LLM-Derived Pedagogical Features : We plan to continue refining the LLM-based feature extraction process. This could involve exploring more sophisticated prompting techniques for features like 'avg_steps', 'level', and 'num_misconceptions', or investigating new types of pedagogical features that LLMs can extract to better capture question complexity.

Human Validation : We will compare LLM annotations for features like cognitive level or misconception identification with human expert annotations for a subset of questions to assess the agreement between automated and human judgments. This validation will help establish the credibility of LLM-extracted features and identify areas where LLM performance could be improved.

By addressing these directions in future research, we aim to strengthen the validity, reliability, and generalizability of our approach to difficulty estimation, potentially establishing a new standard for efficient assessment development in educational contexts.

References

[1] Lawrence Rudner. "Implementing the Graduate Management Admission Test Computerized Adaptive Test". In: Jan. 2010, pp. 151–165. ISBN: 978-0-387-85459-5. DOI: 10.1007/978-0-387-85461-8_8.

[2] James C. Impara and Barbara S. Plake. "Teachers' Ability to Estimate Item Difficulty: A Test of the Assumptions in the Angoff Standard Setting Method". In: *Journal of Educational Measurement* 35.1 (1998), pp. 69–81. DOI: https://doi.org/10.1111/j.1745-3984.1998.tb00528.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1745-3984.1998.tb00528.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1745-3984.1998.tb00528.x.

[3] Georg Rasch. "Probabilistic models for some intelligence and attainment tests". In: *Danish Institute for Educational Research* (1960).

[4] S. AlKhuzaey, F. Grasso, T. R. Payne, et al. "Text-based Question Difficulty Prediction: A Systematic Review of Automatic Approaches". In: *International Journal of Artificial Intelligence in Education* 34 (2024), pp. 862–914. DOI: 10.1007/s40593-023-00362-1.

[5] Inn-Chull Choi and Youngsun Moon. "Predicting the Difficulty of EFL Tests Based on Corpus Linguistic Features and Expert Judgment". In: *Language Assessment Quarterly* 17.1 (2020), pp. 18–42. DOI: 10.1080/15434303.2019.1674315. eprint: https://doi.org/10.1080/15434303.2019.1674315. URL: https://doi.org/10.1080/15434303.2019.1674315.

[6] Le An Ha et al. "Predicting the Difficulty of Multiple Choice Questions in a High-stakes Medical Exam". In: *Proceedings of the Fourteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Ed. by Helen Yannakoudakis et al. Florence, Italy: Association for Computational Linguistics, Aug. 2019, pp. 11–20. DOI: 10.18653/v1/W19-4402. URL: https://aclanthology.org/W19-4402/.

[7] Luca Benedetto et al. "Introducing a framework to assess newly created questions with Natural Language Processing". In: 2020. arXiv: 2004.

13530 [cs.CL]. URL: https://arxiv.org/abs/2004.13530.

[8] Hatice Cigdem Bulut, Okan Bulut, and Serkan Arikan. "Evaluating group differences in online reading comprehension: The impact of item properties". In: *International Journal of Testing* 23.1 (2023), pp. 10–33. DOI: 10.1080/15305058.2022.2044821. eprint: https://doi.org/10.1080/15305058.2022.2044821. URL: https://doi.org/10.1080/15305058.2022.2044821.

[9] Marian Hickendorff. "The Language Factor in Elementary Mathematics Assessments: Computational Skills and Applied Problem Solving in a Multidimensional IRT Framework". In: *Applied Measurement in Education* 26.4 (2013), pp. 253–278. DOI: 10.1080/08957347.2013.824451. eprint: https://doi.org/10.1080/08957347.2013.824451. URL: https://doi.org/10.1080/08957347.2013.824451.

[10] Luca Benedetto et al. *R2DE: a NLP approach to estimating IRT parameters of newly generated questions*. 2020. arXiv: 2001.07569 [cs.LG]. URL: https://arxiv.org/abs/2001.07569.

[11] Joshua F. Lawrence et al. "Reading Comprehension and Academic Vocabulary: Exploring Relations of Item Features and Reading Proficiency". In: *Reading Research Quarterly* 57.2 (2022), pp. 669–690. DOI: https://doi.org/10.1002/rrq.434. eprint: https://ila.onlinelibrary.wiley.com/doi/pdf/10.1002/rrq.434. URL: https://ila.onlinelibrary.wiley.com/doi/abs/10.1002/rrq.434.

[12] Ghader Kurdi, Bijan Parsia, and Uli Sattler. "An Experimental Evaluation of Automatically Generated Multiple Choice Questions from Ontologies". In: Feb. 2017, pp. 24–39. ISBN: 978-3-319-54626-1. DOI: 10.1007/978-3-319-54627-8_3.

[13] Le An Ha and Victoria Yaneva. "Automatic Distractor Suggestion for Multiple-Choice Tests Using Concept Embeddings and Information Retrieval". In: *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Ed. by Joel Tetreault et al. New Orleans, Louisiana: Association for Computational Linguistics, June 2018, pp. 389–398. DOI: 10.18653/v1/W18-0548. URL: https://aclanthology.org/W18-0548/.

[14] Yukie Toyama. "What Makes Reading Difficult? An Investigation of the Contributions of Passage, Task, and Reader Characteristics on Comprehension Performance". In: *Reading Research Quarterly* 56.4 (2021), pp. 633–642. DOI: https://doi.org/10.1002/rrq.440. eprint: https://ila.onlinelibrary.wiley.com/doi/pdf/10.1002/rrq.440. URL: https://ila.onlinelibrary.wiley.com/doi/abs/10.1002/rrq.440.

[15] Zhenya Huang et al. "Question difficulty prediction for READING problems in standard tests". In: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence*. AAAI'17. San Francisco, California, USA: AAAI Press, 2017, pp. 1352–1359. DOI: 10.5555/3298239.3298437.

[16] Zhaopeng Qiu, Xian Wu, and Wei Fan. "Question Difficulty Prediction for Multiple Choice Problems in Medical Exams". In: Nov. 2019, pp. 139–148. ISBN: 978-1-4503-6976-3. DOI: 10.1145/3357384.3358013.

[17] Kang Xue et al. "Predicting the Difficulty and Response Time of Multiple Choice Questions Using Transfer Learning". In: *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*. Ed. by Jill Burstein et al. Seattle, WA, USA → Online: Association for Computational Linguistics, July 2020, pp. 193–197. DOI: 10.18653/v1/2020.bea-1.20. URL: https://aclanthology.org/2020.bea-1.20/.

[18] Luca Benedetto et al. "On the application of Transformers for estimating the difficulty of Multiple-Choice Questions from text". In: *Proceedings of the 16th Workshop on Innovative Use of NLP for Building Educational Applications*. Ed. by Jill Burstein et al. Online: Association for Computational Linguistics, Apr. 2021, pp. 147–157. URL: https://aclanthology.org/2021.bea-1.16/.

[19] Sebastian Gombert et al. "Predicting Item Difficulty and Item Response Time with Scalar-mixed Transformer Encoder Models and Rational Network Regression Heads". In: *Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2024)*. Ed. by Ekaterina Kochmar et al. Mexico City, Mexico: Association for Computational Linguistics, June 2024, pp. 483–492. URL: https://aclanthology.org/2024.bea-1.40/.

[20] Radhika Kapoor et al. *Prediction of Item Difficulty for Reading Comprehension Items by Creation of Annotated Item Repository*. 2025. arXiv: 2502.20663 [cs.CL]. URL: https://arxiv.org/abs/2502.20663.

[21] Giovanni Aradelli. "Transformers for Question Difficulty Estimation from Text". MA thesis. Politecnico di Milano, Dec. 2020. URL: https://hdl.handle.net/10589/170785.

[22] Yixin Liu et al. "Improving Large Language Model Fine-tuning for Solving Math Problems". In: *ArXiv* abs/2310.10047 (2023). URL: https://api.semanticscholar.org/CorpusID:264146593.

[23] Aniket Didolkar et al. "Metacognitive Capabilities of LLMs: An Exploration in Mathematical Problem Solving". In: *ArXiv* abs/2405.12205 (2024). URL: https://api.semanticscholar.org/CorpusID:269921384.

[24] Nicy Scaria, Suma Dharani Chenna, and Deepak Subramani. "Automated Educational Question Generation at Different Bloom's Skill Levels Using Large Language Models: Strategies and Evaluation". In: *Artificial Intelligence in Education*. Springer Nature Switzerland, 2024, pp. 165–179. ISBN: 9783031642999. DOI: 10.1007/978-3-031-64299-9_12. URL: http://dx.doi.org/10.1007/978-3-031-64299-9_12.

[25] Bryan Constantine Sadihin, Hector Rodriguez Rodriguez, and Matteo Jiahao Chen. "[Proposal-ML] Mining Misconception in Mathematics". In: *Submitted to Tsinghua University Course: Advanced Machine Learning*. under review. 2024. URL: https://openreview.net/forum?id=CE85qdNSlp.

Appendix

A Step by Step Example

The following is an example prompt used to generate step by step solution explanations for a mathematical problem:

Prompt

You are an expert in math pedagogy. Your task is to answer the following question deconstructing it to the most elemental steps. Don't skip any step. Don't assume anything about the reader, try to solve it in the most atomic and pedagogical way possible.

Here is the question:

Question: What is the decreasing order of the roots $4\sqrt{3}$, $3\sqrt{4}$, $2\sqrt{5}$, and $5\sqrt{2}$?
A) $2\sqrt{5} > 4\sqrt{3} > 5\sqrt{2} > 3\sqrt{4}$
B) $2\sqrt{5} > 3\sqrt{4} > 4\sqrt{3} > 5\sqrt{2}$
C) $5\sqrt{2} > 3\sqrt{4} > 4\sqrt{3} > 2\sqrt{5}$
D) $5\sqrt{2} > 4\sqrt{3} > 3\sqrt{4} > 2\sqrt{5}$
E) $2\sqrt{5} > 4\sqrt{3} > 5\sqrt{2} > 3\sqrt{4}$
Correct Answer: D

First, think about your task step by step inside <thinking></thinking> tags. Your thinking process should try to go over all the posible ways to solve the question and try to find the most atomic way to solve it.

Then, when you are ready to answer, after your </thinking> tag you should write your response using "Step" as the key and the step as the value. For example:

Thinking process... Think as long as you need to.

Step 1: ...
Step 2: ...
Step 3: ...
...

Important instructions:
- Don't skip any step.
- Don't assume anything about the reader, try to solve it in the most atomic and pedagogical way possible.
- Use as many steps as you need to.

And here is the LLM's response:

Thinking Process

Okay, let's break down this problem. We need to compare four numbers: $4\sqrt{3}$, $3\sqrt{4}$, $2\sqrt{5}$, and $5\sqrt{2}$ and arrange them in decreasing order. The main challenge is that the numbers are in a mixed form (integer multiplied by a square root). To compare them easily, we need to get rid of the square roots or the integers outside the square root.

The easiest way to compare these values is to bring the coefficients *inside* the square root.

Let's consider how to bring a number inside a square root. For example, consider $a\sqrt{b}$. We can rewrite this as $\sqrt{a^2 * b}$.

So, the first step is to convert all the numbers into the form of a single square root.

Then, we simply compare the numbers *inside* the square roots, because if $a > b$, then $\sqrt{a} > \sqrt{b}$.

Finally, we arrange the original numbers based on their corresponding squared values, in decreasing order.

Let's start the solution.

Response

Step 1: Convert $4\sqrt{3}$ into the form $\sqrt{x}$.
We have $4\sqrt{3} = \sqrt{4^2 * 3} = \sqrt{16 * 3} = \sqrt{48}$

Step 2: Convert $3\sqrt{4}$ into the form $\sqrt{x}$.
We have $3\sqrt{4} = \sqrt{3^2 * 4} = \sqrt{9 * 4} = \sqrt{36}$

Step 3: Convert $2\sqrt{5}$ into the form $\sqrt{x}$.
We have $2\sqrt{5} = \sqrt{2^2 * 5} = \sqrt{4 * 5} = \sqrt{20}$

Step 4: Convert $5\sqrt{2}$ into the form $\sqrt{x}$.
We have $5\sqrt{2} = \sqrt{5^2 * 2} = \sqrt{25 * 2} = \sqrt{50}$

Step 5: Now we have $\sqrt{48}$, $\sqrt{36}$, $\sqrt{20}$, and $\sqrt{50}$. We can easily compare the values inside the square roots: 48, 36, 20, and 50.

Step 6: Arrange the values inside the square roots in decreasing order:
50 > 48 > 36 > 20

Step 7: Replace the values inside the square

roots with their original forms:
$5\sqrt{2} > 4\sqrt{3} > 3\sqrt{4} > 2\sqrt{5}$

Step 8: Compare this result with the given options.
The correct option is D: $5\sqrt{2} > 4\sqrt{3} > 3\sqrt{4} > 2\sqrt{5}$.

B Bloom's Cognitive Level Table

Table **??** outlines Bloom's Taxonomy levels used to classify the cognitive demands of problems. Each level is an ordinal number in our feature encoding. The LLM was provided with this rubric and asked to determine the primary cognitive level for each question (feature: `level`).

Cognitive Level	**Description**
Remember	Recall facts and basic concepts
Understand	Explain ideas or concepts, interpret, summarize
Apply	Use information in new situations, execute procedures
Analyze	Draw connections among ideas, break down into parts
Evaluate	Justify a stand or decision, verify, critique
Create	Produce new or original work, design, construct

Table 7: *Bloom's Taxonomy of cognitive levels applied to mathematical problem classification.*

C Example of Misconceptions List

Following the example from the appendix **??**, here is one of the outputs of the LLM when asked about common misconceptions for the question in the example. The prompt for the `num_misconceptions` feature asked the LLM to generate an exhaustive, atomic list of such misconceptions, and the count was averaged over runs.

1. Students may confuse 'decreasing order' with 'increasing order', leading them to reverse the correct order of the numbers.
2. Students might incorrectly apply the property $a\sqrt{b} = \sqrt{a^2 * b}$ by forgetting to square 'a' before multiplying by 'b', calculating it as $a\sqrt{b} = \sqrt{a * b}$.
3. Students might make arithmetic errors when squaring the number outside the square root (e.g., calculating 4^2 as 8 instead of 16).
4. Students might make arithmetic errors in the multiplication step within the square root (e.g., calculating $16 * 3$ as 45 instead of 48).
5. Students may believe that the magnitude of the number outside the square root is the primary determinant of the overall value, even if the value inside the square root is smaller (e.g., incorrectly assuming that $5\sqrt{2}$ is always greater than $2\sqrt{5}$ because 5 > 2).
6. After correctly comparing the values under the square roots, students might forget to convert them back to their original forms when selecting the final answer, leading them to choose an option with the numbers under the square roots ordered.
7. Students might make careless copying errors when transferring the correctly ordered expressions from their intermediate steps to the final answer choice.
8. Students may lack a strong number sense regarding the approximate values of square roots, making it difficult to estimate and compare the values without performing the conversion to a common form.
9. Students may struggle with the concept that irrational numbers can be ordered on a number line, treating them as inherently harder to compare than integers.
10. Students might try to apply the square root individually to each term, incorrectly stating that $a\sqrt{b} = \sqrt{a} * \sqrt{b}$.
11. Students might not understand what a root is and how it affects the order of the numbers.

This was performed 3 times and then averaged the number of misconceptions for each question.

D LLM Feature Extraction Prompts

This section details the core prompts used with Google's Gemini models (primarily `gemini-2.0-flash`, with `gemini-2.5-flash` or `gemini-2.5-pro` for more complex tasks as noted) to extract various pedagogical features. Prompts generally included instructions for chain-of-thought reasoning (within <thinking>...</thinking> tags, not shown in core prompts below for brevity) and a strict JSON output format. Features were typically aggregated over multiple runs (usually 1 or 3) for stability. Question-specific content like `${questionTitle}` and `${optionsJson}` were dynamically inserted.

D.1 Knowledge Dimension (Feature: Knowledge_Dimension)

Model Used: gemini-2.0-flash (1 run)

```
Analyze the following math question.  What
primary type of knowledge does it assess?
Choose one:  - Factual:  Recalling specific
facts, definitions, formulas, or properties.
- Conceptual:  Understanding relationships
between concepts, interpreting information,
explaining ideas, applying knowledge flexibly.
- Procedural:  Executing a sequence of steps or
applying a standard algorithm or method.
Question:  ${questionTitle}
Options:  ${options.map(opt => ‘-
${opt}‘).join(’
n’)}
Think step-by-step about what the student
needs to know or do.  Is it mainly recall,
understanding connections, or following steps?
Then state the single best classification.
```

D.2 Information Gap (Feature: Question_Answer_Info_Gap)

Model Used: gemini-2.5-flash (3 runs, mean aggregated)

```
Analyze the following math question.  How much
information, knowledge, or reasoning *not
explicitly stated* in the question text itself
is required to arrive at the correct solution?
Consider implicit assumptions, required
prior knowledge (beyond basic arithmetic),
or necessary intermediate reasoning steps.
Use this scale:  1 = None:  The answer can be
derived directly using only the information and
numbers explicitly given.  2 = Low:  Requires
minimal prior knowledge (e.g., a common
definition) or a single, obvious implicit
step.  3 = Medium:  Requires recall of specific
formulas/theorems not given, multiple implicit
reasoning steps, or interpretation of context.
4 = High:  Requires significant external
knowledge, complex synthesis of unstated
information, or bridging substantial gaps in
the problem statement.
Question:  ${questionTitle}
Options:  ${options.map(opt => ‘-
${opt}‘).join(’
n’)}
Think step-by-step about the solution process.
What external knowledge (formulas, theorems,
concepts) is needed?  What steps rely on
implicit understanding?  How large is the gap
between the stated information and the required
solution path?
```

D.3 Max Expression Nesting Depth (Feature: Max_Expression_Nesting_Depth)

Model Used: gemini-2.5-flash (3 runs, mode aggregated)

```
Analyze the mathematical expressions in the
following text.  Consider nested parentheses,
functions (like sqrt), fractions, exponents,
etc.  What is the maximum nesting depth
required to parse the most complex part of
any expression?
For example:  - ’2 + 3’ has depth 0.  -
’sqrt(5)’ has depth 1.  - ’3 * (4 + 5)’ has
depth 1.  - ’sqrt(a + (b/c))’ has depth 2
(division inside addition inside sqrt).  -
’((a+b)2̂) / (c - d)’ has depth 2.
Text:  ${questionTitle}
Think step-by-step about the structure of each
expression.  Identify the most deeply nested
part.  Count the levels of nesting.
```

D.4 Most Complex Number Type (Feature: Most_Complex_Number_Type)

Model Used: gemini-2.0-flash (1 run)

```
Analyze the numbers involved in this math
question, including those in the options.
Consider integers, decimals, fractions,
roots/irrationals (like sqrt(2) or pi), and
complex numbers (like 3+2i).  DO NOT consider
abstract variables (like x) for this specific
task.
Based on the following hierarchy of *numerical*
complexity (lowest to highest):  1:  Integer
(e.g., 5, -2, 0) 2:  Decimal (e.g., 3.14,
-0.5) 3:  Fraction (e.g., 1/2, 3/4) 4:
Root/Irrational (e.g., sqrt(3), pi) 5:  Complex
Number (e.g., 2i, 1+i)
What is the HIGHEST level of numerical
complexity present in the question or its
options?
Question:  ${questionTitle}
Options:  ${options.map(opt => ‘-
${opt}‘).join(’
n’)}
Think step-by-step.  Identify the different
types of numbers present.  Determine the highest
level reached according to the hierarchy.
Ignore variables/abstract symbols.
```

D.5 Has Abstract Symbols (Feature: Has_Abstract_Symbols)

Model Used: gemini-2.0-flash (1 run, result to 1/0)

```
Does the following math question or its options
contain abstract symbols representing variables
or unknown quantities (e.g., x, y, a, ?,
width, length)?  Ignore standard mathematical
operators, functions, units, and concrete
numbers.
Question:  ${questionTitle}
Options:  ${options.map(opt => ‘-
${opt}‘).join(’
n’)}
Think step-by-step.  Look for letters or symbols
used as variables.
```

D.6 Units Check (Feature: `Units_Check`)

Model Used: `gemini-2.0-flash` (1 run, result to 1/0)

```
Does the following math question involve units
of measurement (like meters, kg, dollars,
seconds, etc.)  in a way that might require
attention to units or potential conversions to
solve correctly?
Question:  ${questionTitle}
Think step-by-step.  Identify any units
mentioned.  Consider if conversions are needed.
```

D.7 Problem Archetype (Feature: `Problem_Archetype`)

Model Used: `gemini-2.5-pro` (1 run)

```
Classify the primary task or structure of the
following math question.  Choose the single
best fit from this list:  - Word Problem -
Calculation - Equation Solving - Geometric
Reasoning/Proof - Data Interpretation -
Conceptual Definition/Understanding - Formula
Application - Pattern Recognition - Other
Question:  ${questionTitle}
Think step-by-step about what the question asks
the student to *do*.  What is the core task?
```

D.8 Real-World Context Flag (Feature: `RealWorld_Context_Flag`)

Model Used: `gemini-2.0-flash` (1 run, result to 1/0)

```
Does the context of the following math question
primarily involve a real-world scenario (e.g.,
shopping, measuring objects, physics situations,
stories about people) OR is it presented in
a purely abstract mathematical way (e.g.,
manipulating equations, asking about number
properties, abstract geometry without context)?
Respond with either 'REALWORLD' or 'ABSTRACT'.
Question:  ${questionTitle}
Think step-by-step about the setting and
elements described in the question.
```

D.9 Ratio of Abstract to Concrete Symbols (Feature: `Ratio_Abstract_Concrete_Symbols`)

Model Used: `gemini-2.0-flash` (1 run, calculates ratio from counts)

```
Analyze the mathematical content in the text
below.  1.  Count the number of concrete
numerical values (e.g., 5, 3.14, -2, 1/2).
2.  Count the number of abstract symbols
representing variables or unknown quantities
(e.g., x, y, a, ?, width).  Ignore standard
mathematical operators (+, -, =), functions
(sqrt), and units (m, kg).
Text:  ${questionTitle}
Think step-by-step.  List the concrete numbers
found.  List the abstract symbols found.
Provide the counts.
```

D.10 Distractor Plausibility (Features: `LLM_Distractor_Plausibility_Max`, `LLM_Distractor_Plausibility_Mean`)

Model Used: `gemini-2.5-flash` (3 runs, results aggregated for Mean and AvgMax)

```
Analyze the following math question and
its options.  The correct answer is
'${correctOption}'.  For each INCORRECT
option provided below, rate its plausibility
as a potential answer that a student might
mistakenly choose.  Use the following scale:
1 = Very Unlikely:  The option is nonsensical
or completely unrelated.  2 = Unlikely:  The
option results from a significant calculation
error or conceptual misunderstanding.  3 =
Possible:  The option might result from a common
calculation mistake or a minor misunderstanding.
4 = Plausible:  The option is a common mistake
or addresses a known misconception related to
the question.  5 = Very Plausible:  The option
is highly tempting, potentially resulting from a
subtle error or a very common misconception.
Question:  ${questionTitle}
All Options:  ${allOptionsStr}
Correct Answer:  ${correctOption}
Think step-by-step about why each incorrect
option might be chosen or not.  Consider common
student errors for this type of problem.  Then,
provide the rating for each incorrect option.
```