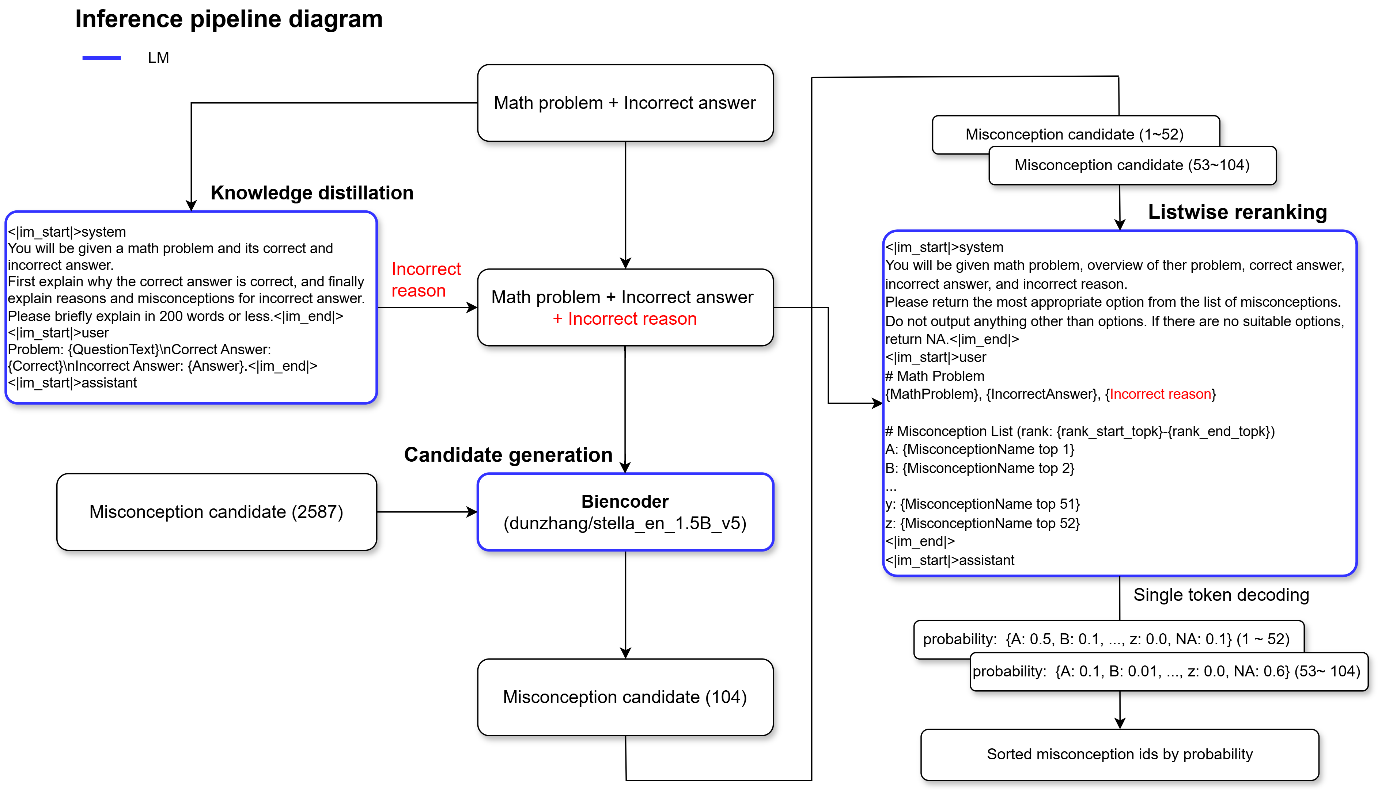
**5th Place Solution**

**Comment**

First, I would like to thank the organizers for hosting such an interesting competition with plenty of room for innovation and no major issues. I also want to thank my teammates who helped create a smooth communication environment for my first team-up experience.

**Solution summary**

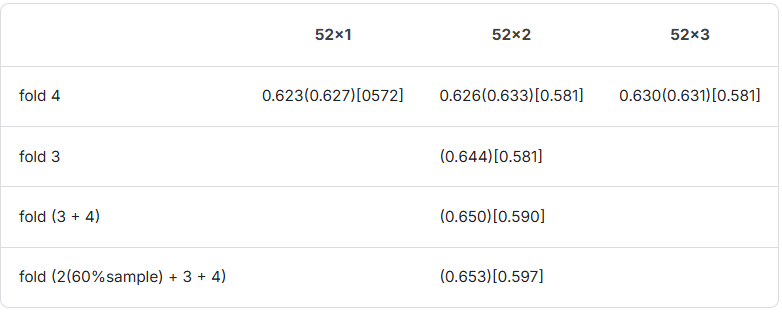


1. **Synthetic data generation**
   * We create problems using LLM with few shot prompting for each MisconceptionId that is not included in the training dataset.
     + For these few shot samples, we insert problems and incorrect answers that have similar MisconceptionNames to the one being generated.
2. **Knowledge Distillation**
   * We input the questiontext, correct answer, and incorrect answer into the LLM (Qwen 2.5 32B Instruct) to generate the reasoning behind the incorrect answer. This content is used as input for all subsequent models (Biencoder, Listwise reranker).
3. **Candidate generation**
   * We use a biencoder model to narrow down the Misconception candidates for each problem and incorrect answer.
4. **Listwise reranking**
   * We input the Misconception candidates sorted by the Biencoder, 52 at a time (corresponding to the number of uppercase and lowercase alphabet letters used as options), into the fine-tuned LLM(Qwen 2.5 32B Instruct) to obtain probability for each option. We then sort the Misconceptions based on these probability.
     + For this process, we use the top 104 candidates from the biencoder. In other words, we perform inference twice with 52 candidates each time, extract the probability, and combine the results.
5. **Ensemble**
   * Since the Listwise reranker uses GPTQ + vLLM inference, a single inference takes only about 120 minutes, allowing the results to be ensembled across three folds. (But due to the random delays discussed [here](https://www.kaggle.com/competitions/eedi-mining-misconceptions-in-mathematics/discussion/550743), our best sub could not include all results from the third fold. I hope that Kaggle staff will address these issues in the future.)

**Detailed solution**

1. **Cross validation**
   * Initially, we used GroupKFold(k=5) based on QuestionId, but since the local cv was significantly lower than lb, we suspected there might be many MisconceptionIds that only appear in the test data. Based on this, we switched to GroupKFold(k=5) using SubjectId. This change resulted in a higher proportion of MisconceptionIds that only appear in validation data, which tended to make the cv closer to lb.
     + example local cv 0.626, lb: 0.633
2. **Synthetic data generation**
   * Synthetic data generation is essential to include MisconceptionIds without existing problems in the prediction results. For each MisconceptionName without problems, we generated one set of existing dataset information (QuestionName, SubjectName, ConstructName, Correct Answer, Incorrect Answer) using few shot prompting (4 shots) with LLM (gemini-1.5-pro).
     + For the few shot prompting samples, we embedded all MisconceptionNames using the model (dunzhang/stella\_en\_1.5B\_v5) and included problems corresponding to similar MisconceptionNames. This resulted in approximately +0.01 improvement in cv compared to using random samples.
3. **Knowledge Distillation(KD)**
   * We input the QuestionText, correct answer, and incorrect answer into the LLM (Qwen 2.5 32B Instruct) to estimate the Misconception. Using this information improves the biencoder's cv+0.04. While the Listwise reranker mentioned later uses the same model as KD, it showed an increase of about cv+0.01 compared to when it's not included.
     + prompt
   * PROMPT\_FORMAT = """<|im\_start|>system
   * You will be given a math problem and its correct and incorrect answer.
   * First explain why the correct answer is correct, and finally explain reasons and misconceptions for incorrect answer.
   * Please briefly explain in 200 words or less.<|im\_end|>
   * <|im\_start|>user
   * Problem: {QuestionText}\nCorrect Answer: {Correct}\nIncorrect Answer: {Answer}.<|im\_end|>
   * <|im\_start|>assistant
   * """
4. **Candidate generation**
   * We extract MisconceptionName candidates for each problem and incorrect answer pair using Biencoder. While we perform negative mining, hard negatives were not effective, so we include slightly relaxed samples.
   * Training parameters
     + library: SentenceTransfomer
     + model: dunzhang/stella\_en\_1.5B\_v5
     + negative mining params
       - range\_min: 512
       - num\_negatives: 2
     + lora config
       - r: 32
       - lora\_alpha: 64
     + loss: CachedMultipleNegativesRankingLoss
     + epoch: 1
   * cv: 0.41, lb: 0.39
5. **Listwise reranker**
   * Overview
     + The reranking part using LLM appears to be the most unique aspect of this competition, and I expect that how this part was implemented significantly impacted the scores. Our teammate [@kcotton21](https://www.kaggle.com/kcotton21) had already come up with the idea of generating single tokens and using their logits for sorting at the beginning of the competition (reference: <https://arxiv.org/abs/2406.15657)>, and our solution is based on this.
   * prompt
     + After various trial and error, we ultimately performed regular fine-tuning using the following prompt to have the LLM generate appropriate choices (which must be single tokens to enable sorting by logits(probability)).
   * NA\_PROMPT\_FORMAT = """<|im\_start|>system
   * You will be given math problem, overview of ther problem, correct answer, incorrect answer, and incorrect reason.
   * Please return the most appropriate option from the list of misconceptions. Do not output anything other than options. If there are no suitable
   * options, return NA.<|im\_end|>
   * <|im\_start|>user
   * # Math Problem
   * Problem: {QuestionText}\nOverview: ({SubjectName}){ConstructName}\nCorrectAnswer: {Correct}\nIncorrectAnswer:
   * {Answer}\nIncorrectReason: {kd}
   * # Misconception List (rank: {rank})
   * {misconception\_list}<|im\_end|>
   * <|im\_start|>assistant
   * {label\_choice}"""
     + misconception\_list
       - It contains 52 choices of uppercase + lowercase alphabets and their corresponding MisconceptionNames. They are ordered by the Biencoder output.
       - A: Thinks sign for parallel lines just means opposite
       - B: Does not recognise the notation for parallel sides
       - .
       - .
       - .
       - y: Thinks that co-interior angles can lie on opposite sides of a transversal
       - z: Believes the gradient of perpendicular lines just have opposite signs<|im\_end|>
     + label\_choice
       - This is the correct answer choice used for fine-tuning. If the correct answer is not included in the choices, "NA" (which is also a single token in Qwen 2.5 32B) is inserted.
     + rank
       - Indicates which range of choices {start\_idx - end\_idx} from the Biencoder's output is included.
   * Training
     + We divided the top 208 outputs from the biencoder into non-overlapping groups of 52 each (1-52, 53-104, 105-156, 157-208) and performed fine-tuning.
     + Training parameters
       - loss
         * This is the standard cross-entropy loss used in regular fine-tuning, calculated across the entire vocabulary. We chose this simpler implementation method since calculating cross-entropy limited to only the candidate choices + "NA" token showed little difference in results.
       - model: Qwen/Qwen2.5-32B-Instruct
       - lora\_config
         * r: 24
         * lora\_alpha: 48
       - lr: 5e-5
       - scheduler: cosine
       - warmup\_steps: 10
       - epoch : 1.0
       - global batch size: 8
       - library: unsloth
   * Inference
     + From the biencoder output, we take the top 104 items and split them into two sets (1-52 and 53-104). Each set is input to the model, generating only single tokens to obtain probabilities for each option. The results from both sets are then combined and sorted.
       - While adding to make it top 156 (52x3 inference) showed a slight improvement in local CV, it wasn't reflected in the leaderboard score, so we didn't adopt this approach.
     + Quantized to 4-bit using GPTQ and inference is performed using vLLM

**cv(public)[private] scores**



**Computational costs**

My local machine is 1 x RTX 4090.

I rented an A100 (from [vast.ai](http://vast.ai/)) for training a 32B model, and including experiments, it cost me about $350 in total…

Training one 32B Listwise model took about 7 hours on the A100.

**What didn't work**

* Options used in the Listwise reranker
  + As single token choices, we tried combinations of two alphabets, combinations of alphabets and numbers 0-9, and japanese hiragana, but all of these approaches performed worse than using just the 52 alphabets.
* utilization of QwQ-32B-preview
* multi step reraking (Qwen 2.5 7B(104 candidate) → Qwen 2.5 32B(52 cadindate))
* adding examples of problems and misconceptions in the reraking prompts.

**Code release**

We released training code. → [Eedi-5th-solution](https://github.com/ebinan92/Eedi-5th-solution?tab=readme-ov-file)