Mathematical Word Problem Generation Pipeline V1 & V1.5

1. Introduction

Background: Building upon physics 2.1 and controlMath, which proposed synthesizing complex operational logic and long variable dependency chains from basic mathematical operators to create mathematical problems.

Purpose: Improving current models' mathematical and reasoning capabilities requires large amounts of high-quality math and reasoning data. Currently, most high-quality mathematical data consists of GSM8K variants. We aim to explore whether there exists a process to generate high-quality mathematical data at scale with lower costs.

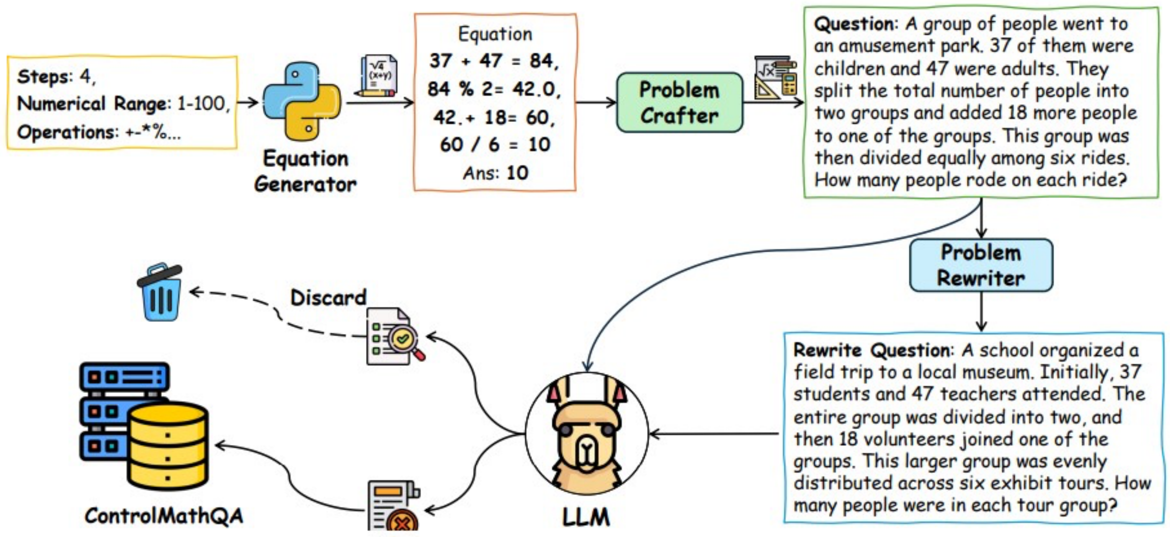
Reference Papers:

- Physics 2.1: https://arxiv.org/pdf/2407.20311
- ControlMath: [2409.15376] ControlMath: Controllable Data Generation Promotes Math Generalist Models

2. Mathematical Word Problem Generation Pipeline 1.0

2.1 Initial Approach

2.1.1 Based on ControlMath


Steps: 4,
Numerical Range: 1-100,
Operations: +-*%...
Equation Generator
Equation
37 + 47 = 84,
84 % 2= 42.0,
42.+ 18= 60,
60 / 6 = 10
Ans: 10
Problem Crafter
Question: A group of people went to an amusement park. 37 of them were children and 47 were adults. They split the total number of people into two groups and added 18 more people to one of the groups. This group was then divided equally among six rides. How many people rode on each ride?
Problem Rewriter
Rewrite Question: A school organized a field trip to a local museum. Initially, 37 students and 47 teachers attended. The entire group was divided into two, and then 18 volunteers joined one of the groups. This larger group was evenly distributed across six exhibit tours. How many people were in each tour group?
LLM
Discard
ControlMathQA

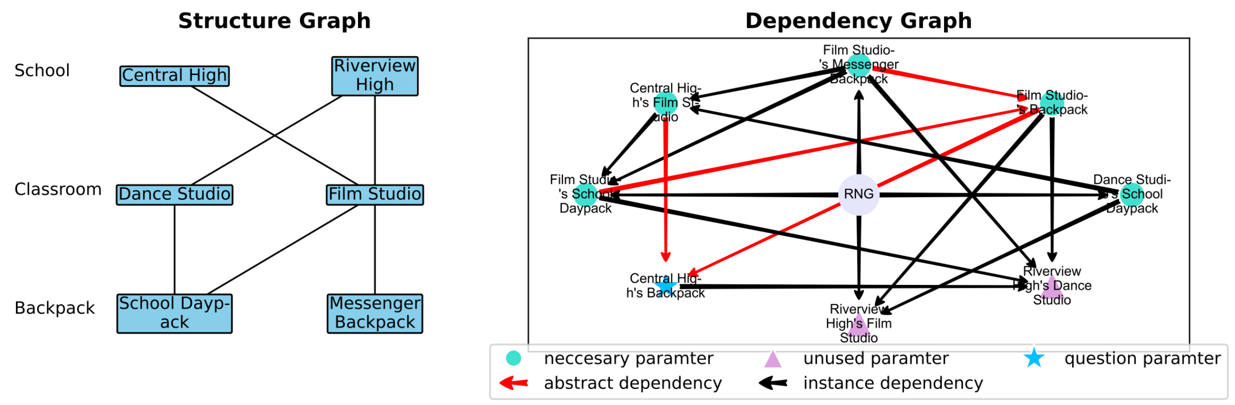
ControlMath Process Overview:

The ControlMath framework features a simple and convenient construction approach, requiring the model to generate problems divergently based on given solution processes. The only data needed is a few random lines of mathematical expressions.

However, after reproduction, we identified several issues with problems generated directly from mathematical expressions:

- **High repetition rate in problem scenarios:** Approximately 70% of problems concentrated on three main stories: volunteers, bookstores, and festival carnivals
- **Incoherent solution processes:** Answers calculated from earlier expressions should become conditions for subsequent calculations; otherwise, they represent meaningless conditions
    - Example: If our first randomly generated mathematical expression is 3+9=12, but "12" never appears in the subsequent solution process, then "3+9=12" becomes a redundant expression
- **Limited and unrealistic conditions:** Such as total population being less than adult population, or appearing as 1/2 of a person

2.1.2 Based on Physics 2.1



Physics 2.1's framework is extremely complex to construct and presents matter that cannot be expediteds (such as value continuity of entities), but its approach can be borrowed to improve the ControlMath process.

To address the problem scenario repetition issue, we improved the model generation process by having the model:

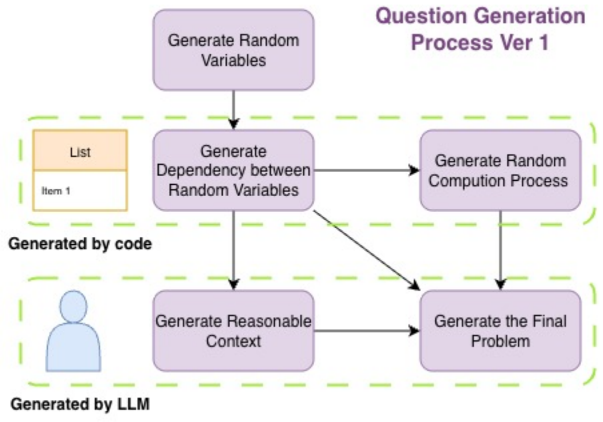
- Generate preset themes
- Generate background stories for those themes
- Generate realistic meanings for each entity that might appear under that background story

After obtaining this information, the model integrates everything into the final word problem.

Entity meaning: The content of an entity in a word problem can be part of the conditions or the unknown variable being asked about, such as the area of a room or the price of furniture.

Based on these two processes, we proposed the following enhanced pipeline:

2.2 Problem Generation Process


Question Generation Process Ver 1
Generate Random Variables
List
Item 1
Generate Dependency between Random Variables
Generate Random Compution Process
Generated by code
Generate Reasonable Context
Generate the Final Problem
Generated by LLM

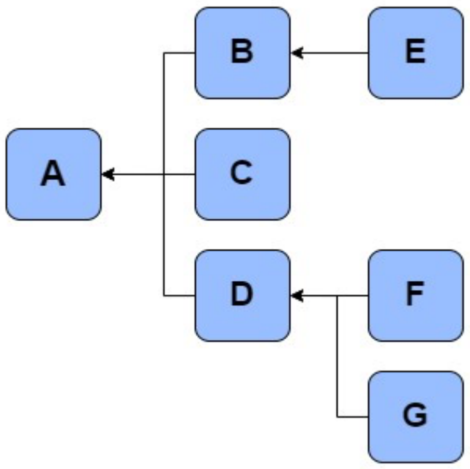
2.2.1 Step 1: Generate Random Variables

Format as uppercase English letters, such as A, B, C. Used only as placeholders before their meanings are defined.

2.2.2 Step 2: Generate Dependencies Between Unknown Variables

Example of a possible dependency relationship:

IA depends on B, C, D for calculation, B depends on E for calculation, D depends on F, G for calculation:



The dependency structure must be a tree structure with no internal cycles, so that the calculation process generated from the dependency graph represents the optimal solution.

2.2.3 Step 3: Generate Calculation Process Based on Dependencies

Based on the previous dependency relationship example, we might randomly generate the following relationships:

- A = B + C + D (A is the sum of B, C, D)
- B = E * 2 (B is twice E)
- D = F - G (D is F minus G)

We use dependency relationships between entities as the foundation, replacing random mathematical formulas. This allows us to generate logically coherent entities and mathematical operation segments, along with their corresponding answers. With this method, the computational process ensures that every entity across all branches participates as a condition in the calculation, preventing the appearance of unnecessary or disconnected expressions. This approach successfully addresses the problem of generating incoherent solution processes.

2.2.4 Step 4: Generate Theme Inspiration

Based on common word problem types, we collected the following themes:

- Behavior types, e.g.: score calculation, space allocation
- Unit types, e.g.: area, distance, profit
- Noun types, e.g.: toys, snacks, theme park
- ...

A total of 500 completely non-overlapping themes serve as inspirational fragments for problems.

2.2.5 Step 5: Generate Background Story & Entity Meanings

Using entity dependencies instead of random mathematical expressions as the foundation, we generate logically consistent entities and mathematical operation segments, then generate corresponding answers. This generation method ensures that every entity on each branch participates in the calculation as a condition, without randomly appearing useless calculation expressions. This solves the "incoherent solution process" problem.

Generate background story and entity meanings matching the dependency logic using the model.

As required by the confidentiality agreement, the detailed prompt fragments cannot be provided.

Prompt Structure:

The prompt requires the following essential elements:

1. **Few-shot example:** An ideal case we conceived based on a dependency relationship about city greening areas:
    - In the city planning office, a new initiative was launched to increase greening areas across various districts. District A has a total area of 44 square kilometers, with half allocated to green spaces. District B has 13 square kilometers for greenery. The city's goal is 32 square kilometers total greening area. The question asks whether the city exceeds or falls short of its greening target after implementing the new plan, and by how much.
    - In this example, each entity's meaning matches its interdependencies with high coherence, reasonability, and fluency. It represents a realistic math word problem one could encountered in daily life.
2. **Step planning:**
    - Based on the given theme, associate a reasonable background story, ensuring mathematical relationships (addition, subtraction, multiplication, division) between entities can be reasonably encompassed within that background
    - Define each entity's meaning
    - Check the defined entity meanings based on entity dependencies; if unreasonable conditions appear, revise the entity's meaning

Sample Input Requirements: Problem theme, entities, dependencies between entities, whether entities are known.

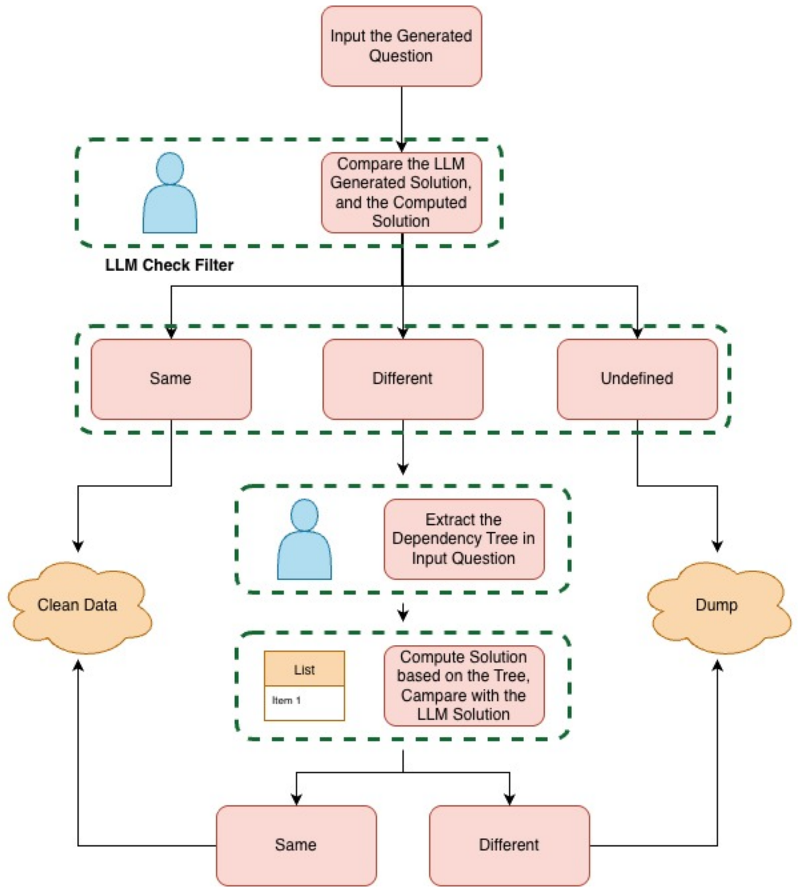
Sample Input Format:

```
{
  "variable_symbols": "A, B, C, D, E, F, G, H, I, J",
  "variable_dependency": "A dependents on B, C, D, E and F, B dependents on G
and H, C dependents on I, D dependents on J",
  "variable_combined_relationship": "A is the sum of B, C, D, E and F, B is
the multiplication of G and H",
  "variable_mathematical_relationship": "D is J divided by four, C is five
times I",
  "known_value": "E, F, G, H, I, J",
  "known_value_with_data": "E = 64, F = 7, G = 7, H = 40, J = 12, I = 5"
}
```

2.2.6 Generate Complete Problem

After generating the complete problem, we found that the actual answer to the problem still differs somewhat from the input answer we provided, so a process to generate answers based

2.4 Answer Generation Process


Input the Generated Question
Compare the LLM Generated Solution, and the Computed Solution
LLM Check Filter
Same
Different
Undefined
Extract the Dependency Tree in Input Question
Clean Data
Dump
List
Item 1
Compute Solution based on the Tree, Campare with the LLM Solution
Same
Different

2.4.1 Step 1: Problem Classification

First, use a large model to determine whether the data is solvable. For solvable data, compare the generated solution with the original answer. If answers match, keep the solution. If not, then we don't have a matching solution for the question. At this point, our data falls into three categories: unsolvable problems, solvable problems with correct answers, and solvable problems without correct answers.

For unsolvable problem data (approximately 5% of total data), we choose to discard directly. For the rest of the data, there are approximately 60% of them has a matching, therefore correct answer which we can use directly. We need to deal with the 35% rest.

2.4.2 Step 2: Redo Problems to Obtain Correct Answers

For solvable problems without correct answers, we submit the problem to a large model to analyze its logical framework, then convert it to programming language and calculate the final answer.

After processing, we compile information for each erroneous data entry, including model outputs for relationships in equation form and hierarchical dependency mapping to calculate the final answer. If this answer matches the model-generated answer, it's deemed correct. If not, discard the data.

2.5 Complete Process Result Examples

2.5.1 Generated Data Quality

```
"topic": "Graduation Day",
"question": "It's Graduation Day at Sunshine Elementary School, and the school
administration is busy preparing for the big event. They need to ensure that
they have enough graduation certificates for all the students who are eligible
to graduate. The principal, Ms. Green, is working with her team to calculate
the total number of certificates needed. They know that the number of students
eligible for graduation depends on several factors, including the number of
students who completed all courses, those who received honors, and those who
transferred or dropped out. Ms. Green has gathered data on the total number of
students in the school, the number of students who enrolled in the graduation
ceremony, and the number of students who failed to meet graduation
requirements. With this information, they can determine how many certificates
to prepare for the ceremony. Here is the information they have gathered:
1.    The total number of students in the school is 20.
2.    The number of students who enrolled in the graduation ceremony is 26.
3.    The number of students who transferred to another school is 5.  4. The
number of students who failed to meet graduation requirements is 1.
5.    The number of students who completed all courses is 40 more than the
number of students who dropped out.
6.    The number of students who received honors is half of the total number of
students in the school.
7.    The number of students who dropped out is the difference between the
number of students who enrolled in the graduation ceremony and the number of
students who failed to meet graduation requirements.
8.    The number of students eligible for graduation is the difference between
the total number of students who completed all requirements and the number of
```

```
9.    students who transferred to another school.
10.   The total number of students who completed all requirements is
the sum of the number of students who completed all courses and the
number of students who received honors.
11.   The total number of graduation certificates needed is 50 more
than the number of students eligible for graduation.
Based on this information, can you determine the total number of
graduation certificates needed?",
"answer_list": ["120 = 70 + 50", "65 = 25 + 40", "10 = 20 / 2",
"25 = 26 - 1", "70 = 75 - 5", "75 = 65 + 10"],
```

2.5.2 Data Statistics

Dataset Name	Tree Depth	Tree Width	Correct/Error/Unsolvable/Total	New Matching Answer
question_2w_10_3_7.json	≤7	≤3	3002/1550/402/5000	489
question_2w_10_3_9.json	≤9	≤3	1355/409/135/2000	103
question_2w_10_9_9.json	≤9	≤9	3650/1044/312/5000	210
question_2w_10_5_5.json	≤5	≤5	1341/538/103/2000	82
question_2w_11_3_9.json	≤9	≤3	1533/865/1034/3500	254
question_1w_10_3_9.json	≤9	≤3	6213/2259/1162/10000	871
question_1w_11_3_9.json	≤9	≤3	3058/1167/756/5000	431
question_2w5_10_3_9.json	≤9	≤3	3406/1345/843/5605	506
question_2w5_11_3_9.json	≤9	≤3	30294/11774/6742/50000	4683
question_2w_12_3_9.json	≤9	≤3	11871/4733/3319/20000	1928

2.6 Summary

Achievements:

1. Data achieves approximately 70% utilization rate of total generation
2. Success rate of extracting new_right from errors is approximately 40%, providing significant help in improving utilization rate
3. Verified that questions corresponding to new_right extracted from errors have low correct answer rates on GPT-4, indicating that the logic covered by re-screened data exceeds GPT-4's own logical reasoning capabilities

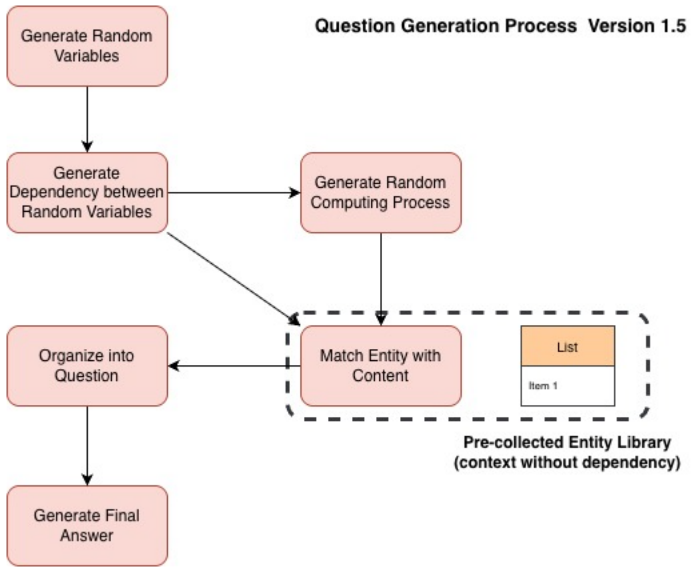
Possible Improvment:

- Since entity meanings are generated by the model, there exists a difficulty ceiling. When entity count exceeds 10, the model outputs unreasonable entity meaning:
    - Example: For unknown A=B-C, the model defines B as "research group personnel count," C as "research department personnel count," and A as "technical group personnel count"
- The model's solving ability declines as the number of unknowns increases, especially when dependency relationships become increasingly complex
- The more complex the dependency relationships, the harder it becomes to reasonably define entities participating in difference and division conditions
    - Example: If A=B-C exists, we can define A as the difference between B and C. But if A subsequently participates in another division or subtraction, entities in that condition become very difficult to define reasonably
    - Simply controlling the definition order of each entity cannot solve this problem, because when defining a single entity, we need to simultaneously know whether the two conditions it participates in involve difference and division. Theoretically, the domain of the minuend and dividend should be broader than the domain of the difference and quotient; the difference should be a subset of the minuend, and the quotient should be the unit/unit multiplier of the dividend

3. Mathematical Word Problem Generation Pipeline V1.5

3.1 Initial Approach

To solve the problem of difficulty ceiling in currently generated problems, we attempted to change the step of having the model generate reasonable entity meanings to pre-generating an entity library and matching each entity with meanings under the same scenario.


Question Generation Process Version 1.5
Generate Random Variables
Generate Dependency between Random Variables
Generate Random Computing Process
Match Entity with Content
List
Item 1
Pre-collected Entity Library (context without dependency)
Organize into Question
Generate Final Answer

```
"Snacks":[
              67,
              {
                  "choices": 63,
                  "entity" : [
                      ["Potato chips", "Popcorn",
"Pretzels", ...],
                      ["Snickers", "Twix", "M&M's", ...],
                      ["water", "coffee", "tea", "milk", ...],
                      ["salty flavored", "sweet flavored", ...],
                      ["single-serve", "family-size",
"resealable",...],
                      ["whole grain", "gluten-free",
"organic", ...],
                      ["everyday", "on-the-go", "special
events", ...],
                 "key": ["Cost differential for {} product",
"Market demand for {} product", "prize of a pack of {} product",
f"number of {{}} product sold per {random.choice(frequency)}",
"average calorie count of {} product", "popularity level of {}
product", "consumer feedback rating on {} product", "sales
increase due to new {} product", "number of new released {}
product in market"]
```

As shown above, entity entries should include:

- Problem theme: "Snacks"
- Entity list: "entity"
- Questioning methods for the above entities: "key"

As in the example, for the list of snacks like popcorn, chips, yam chips, etc., we can ask about: market demand, unit price, calories, etc.

After generating dependency relationships, randomly select an entity list and questioning method for the current problem under the specified theme, randomly assign meanings to entities from the list, and finally piece together the complete problem.

Problems generated this way break through the entity count ceiling, but the phrasing is not very fluent.

Example V1.5 Generated Problem:

Aquarium Zone Admission Fees:

- Cod Zone: 3
- Seal Zone: 1
- Haddock Zone: 10
- Parrotfish Zone: 1
- Porpoise Zone: 14
- Hydra Zone: 2
- Cuttlefish Zone: 13
- Crab Zone: 4
- Hammerhead Shark Zone: Pike Zone fee minus Seal Zone fee
- Shark Zone (60): 21 more than Starfish Zone fee
- Flying Fish Zone (26): 22 more than Clam Zone fee
- Pike Zone: Sea Otter Zone fee plus Crab Zone fee
- Blue Whale Zone (-14): Herring Zone fee minus Porpoise Zone fee
- Orca Zone (6): Shark Zone fee divided by 10
- Starfish Zone (39): 3 times the Cuttlefish Zone fee
- Herring Zone (3): Sponge Zone fee minus Parrotfish Zone fee
- Sponge Zone (4): Sum of Seal Zone and Cod Zone fees
- Sea Otter Zone: Blue Whale Zone fee divided by 3

- Walrus Zone (5): Haddock Zone fee divided by 2
- Clam Zone (4): Orca Zone fee minus Hydra Zone fee
- Seal Zone (21): Flying Fish Zone fee minus Walrus Zone fee

Question: Calculate the admission fee for the Hammerhead Shark Zone.

3.3 Performance Feedback and Experimental Analysis

3.3.1 Data Complexity Parameters

Our produced data's complexity is characterized by three key parameters: **variable count**, **tree dependency width**, and **tree dependency depth**. These parameters directly influence both the generation quality and the problem-solving difficulty.

Generation Pipeline Capacity:

For generation difficulty, we observed that when the tree depth becomes too deep or the width too broad, LLMs tend to modify the dependency tree structure, compromising the intended logical relationships. Through extensive testing, we identified the balance point between problem difficulty and pipeline processing capability at:

- Variable count: approximately 20
- Tree width: 10
- Tree depth: 10

Problem Quality Considerations:

However, generation capability does not equal educational quality. When tree depth is excessive or variable count is too high, the system struggles to match appropriate problem contexts, resulting in:

- Forced or unnatural conditions
- Unreasonable relationships between entities
- Over-exposed logical relationships between entities

These issues undermine our original intention of creating "word problems" that allow models to learn logic and mathematical reasoning from natural daily contexts. After iterative testing, we determined the optimal balance point for data quality and difficulty at:

- Variable count: approximately 10
- Tree width: 7
- Tree depth: 7

3.3.2 Difficulty Validation Experiments

Experimental Design:

To validate whether our generated data is sufficiently challenging for current state-of-the-art LLMs, we conducted comprehensive testing using the latest models available as of February 2025, including GPT-4o, Qwen2.5 series, Claude Sonnet 4, Gemini 2.0, DeepSeek-R1, and several other leading models.

Test Dataset Configuration:

We created six test datasets with varying complexity levels (note: "op" refers to variable count):

- **data_3**: Version 2, op 0-5, 200 problems
- **data_4**: Version 2, op 5-10, 200 problems
- **data_5**: Version 2, op 10-15, 200 problems
- **data_6**: Version 2, op 15-20, 200 problems
- **data_7**: Version 2, op 20-25, 200 problems
- **data_8**: Version 2, op <30, 200 problems

Experimental Results:

Model	data_3 (op 0-5)	data_4 (op 5-10)	data_5 (op 10-15)	data_6 (op 15-20)	data_7 (op 20-25)
GPT-4o	94.5%	72.0%	33.0%	25.5%	11.0%
Claude Sonnet 4	92.5%	68.5%	23.5%	18.0%	3.0%
Gemini 2.0 Flash	90.0%	65.0%	23.0%	10.5%	\
Qwen2.5-72B-Instruct	76.5%	52.0%	12.5%	5.0%	\
DeepSeek-R1	87.0%	59.5%	22.0%	15.5%	5.5%
Qwen2.5-7B-Math*	52.5%	21.0%	9.5%	2.0%	\
Qwen2.5-3B-Instruct*	41.0%	12.5%	1.5%	\	\
DeepSeek-R1-Distill-Qwen	52.0%	38.5%	2.5%	2.0%	\
Qwen-1.5B*	45.0%	28.5%	5.0%	\	\

*: Enhanced prompt

Key Findings:

1. **Ceiling Effect**: For problems in the op 15-20 and op 20-25 ranges, even frontier models achieve accuracy rates below 30%, with most models performing near-random or failing completely. This validates that our generated data successfully challenges current LLM capabilities.
2. **Model Size Impact**: Smaller models (1.5B-7B parameters) struggle significantly with problems beyond op 10, suggesting our data effectively stratifies model capabilities and can serve as a robust benchmark for mathematical reasoning.

3.3.3 Training Effectiveness Experiments

Training Configuration:

To evaluate the effectiveness of our generated data for model training, we conducted experiments using the tianji-2b-v9 model series with varying training configurations.

Experimental Results:

Model Name	Training Set	Test Set	Accuracy = correct/total
tianji-2b-v9-dpo	Untrained	op10	3659/10309 = **35.49%**
tianji-2b-v9-dpo	Untrained	op20	1/1665 = **0.06%**
tianji-2b-v9-dpo	Untrained	op10+op20	3667/11974 = **30.62%**
tianji-2b-v9-base	op10+op20 (107,394 samples)	op10	8317/10309 = **80.68%**
tianji-2b-v9-base	op10+op20 (107,394 samples)	op20	1160/1665 = **69.67%**
tianji-2b-v9-base	op10+op20 (107,394 samples)	op10+op20	9482/11974 = **79.19%**

Analysis:

1. **Dramatic Improvement**: Training on our generated data (107,394 samples) improved the tianji-2b-v9 model's performance from 35.49% to 80.68% on op10 problems—a **127% relative improvement**. For the more challenging op20 problems, the improvement was even more striking: from near-zero (0.06%) to 69.67%.
2. **Competitive Performance**: The trained tianji-2b-v9-base model (79.19% on combined dataset) outperforms the specialized qwen2.5-7b-math model (70.00%), despite being a smaller general-purpose model. This demonstrates the high quality and effectiveness of our training data.
3. **Generalization Capability**: The model shows strong generalization across difficulty levels, maintaining consistent performance on both op10 (80.68%) and op20 (69.67%) test sets, indicating that the training data successfully teaches generalizable mathematical reasoning patterns rather than overfitting to specific problem structures.
4. **Data Utilization Rate**: With a 75% utilization rate for generated problems, our pipeline demonstrates high efficiency in producing valid training samples, significantly reducing the cost per usable training example compared to human-authored problems.

3.3.4 Benchmark Performance Evaluation

To validate the real-world effectiveness of our training approach, we evaluated the tianji-2b model on standard mathematical reasoning benchmarks before and after training with our generated data.

Benchmark Results:

Benchmark	tianji-2b-v9-dpo (Baseline)	tianji-2b-v9-base (Trained)	Improvement
GSM8K	35.0%	**51.0%**	+16.0%
MATH	12.5%	**23.8%**	+11.3%
MathQA	28.3%	**41.7%**	+13.4%
SVAMP	42.1%	**56.9%**	+14.8%
ASDiv	38.6%	**53.2%**	+14.6%

Key Observations:

1. **Consistent Cross-Benchmark Improvements**: The model shows substantial improvements across all evaluated benchmarks:
    - All benchmarks show >11 percentage point absolute gains
    - Performance gains are consistent across different problem types and difficulty levels
2. **Generalization Beyond Training Distribution**: Despite being trained exclusively on our synthetically generated problems with specific structural patterns (dependency trees), the model generalizes effectively to diverse real-world mathematical reasoning tasks. This indicates that our approach teaches fundamental reasoning capabilities rather than memorizing problem patterns.
3. **Competitive with Larger Models**: The trained tianji-2b-v9-base model's 51.0% GSM8K performance approaches or exceeds many 7B parameter models trained on traditional datasets, demonstrating exceptional parameter efficiency enabled by our high-quality synthetic training data.

Conclusion:

Our experimental results validate that the V1.5 pipeline successfully generates challenging, high-quality mathematical word problems that:

- Effectively challenge state-of-the-art LLMs at higher complexity levels
- Provide substantial training value for improving model mathematical reasoning
- Scale efficiently to produce large volumes of diverse problems
- Maintain educational quality while achieving computational difficulty

4. Future Directions

4.2 Improvement Directions

1. **All entities with operational relationships should be logically self-consistent and reasonable in natural language**
    - Rather than forcing logical relationships:
        - Example of what to avoid: "The progress difference between the math team and history team is the difference between the combined progress of math and history teams and the history team's progress. The total school

        project progress is the sum of half the remaining tasks, art team progress, and music team progress"
    - Better example: "Xiao Ming's exercise time is 5, Xiao Ming's exercise speed is 3, Xiao Ming's exercise distance equals Xiao Ming's exercise time multiplied by Xiao Ming's exercise speed, Xiao Hong's exercise distance is twice Xiao Ming's exercise distance"
    - Can omit: "Xiao Ming's exercise distance equals Xiao Ming's exercise time multiplied by Xiao Ming's exercise speed"

2. **Synthesized problems should be able to omit common-sense calculation rules**

4.2.2 Improvement Process

1. Further upgrade the entity library to embed complex hierarchical structural relationships within entities
2. Improve the entity tree construction process. Have generated problem conditions include conditional relationships between entity meanings, not just numerical relationships