

T2 多样性数据生成与相似度计算

故事背景

你正在为一个在线学习平台开发智能题库系统。为了让学生能够充分练习，题库需要满足两个核心要求：

- **准确的相似度判断**：能够识别题目之间的相似程度，避免重复练习相同的题目
- **多样化的题目生成**：能够生成大量风格各异的题目，让学生接触到不同的表达方式

然而，题目的相似度判断并不简单。同样是 $12 + 35$ 这道题，可能会有很多种表达方式：

- 符号变体： $12 + 35$ 、 $12 \oplus 35$
- 顺序变化： $35 + 12$
- 文字表达： 12 加 35 、计算： 12 加 35

你需要实现一个能够准确计算文本相似度的函数，并基于此生成一批高质量、多样化的数学题目数据集。

任务目标

你需要完成两个核心任务：

1. 实现相似度计算：`compute_similarity(text1, text2, model, tokenizer) -> float`

编写一个函数，使用给定的 embedding 模型计算两个文本的余弦相似度。

技术要点：

- Last-token pooling (提取最后一个 token 的 hidden state)
- L2 normalization (归一化向量)
- 余弦相似度 (归一化后的点积)

2. 生成多样化数据集：`data/dataset.jsonl`

生成 1024 条高质量的数学题目数据，满足以下要求：

- 所有题目唯一（无重复）
- 每个题目最多 3 个数字
- 答案为 3 位数（100-999）
- 平均相似度 ≤ 0.5 （越低越好）

实现要求

文件结构

Plain Text

T2/

```
|— data/
|   |— test_data_1.jsonl      # 相似度测试用例（10 组）
|   |— dataset.jsonl        # 考生生成的数据集（1024 条）
|— evaluate.py               # 评测主程序（不可修改）
|— submission.py             # 考生实现文件（仅此文件可修改）
|— README.md                 # 本文档
```

修改规则

- ✓ 仅可修改 `submission.py` 中的 `compute_similarity` 函数
- ✓ 可以新增辅助函数用于数据生成
- ✓ 必须生成 `data/dataset.jsonl` 文件（1024 条数据）
- ✗ 不得修改 `evaluate.py` 的任何内容
- ✗ 不得修改 `data/test_data_1.jsonl` 测试用例

技术要求

相似度计算：

- 使用 PyTorch 和 Transformers 库
- 正确实现 last-token pooling
- 正确实现 L2 normalization
- 返回值范围：[0, 1]

数据生成：

- JSON Lines 格式：每行一个 JSON 对象
- 必需字段：problem (题目)、answer (答案)
- 题目示例：{"problem": "12 + 35", "answer": "47"}

评测说明

运行方式

演示模式（详细输出）：

```
Bash
python evaluate.py
# 或
python evaluate.py --mode demo
```

- 展示每个测试用例的详细结果
- 显示数据多样性的详细统计
- 显示扣分明细和最终得分

评分模式（简洁输出）：

```
Bash
python evaluate.py --mode grading
```

- 简洁的进度输出
- 最终输出详细评分
- 用于正式评测

评测流程

第一部分：相似度计算准确性（50 分）

测试内容：

- 10 组预定义的文本对测试用例
- 使用预计算的标准相似度进行对比
- 测试模型对各种相似度的判断能力

示例测试用例：

JSON

```
{"text1": "12 + 35", "text2": "12 + 35", "description": "完全相同",  
"standard_similarity": 1.0}  
{"text1": "12 + 35", "text2": "12+35", "description": "符号不同",  
"standard_similarity": 0.877}  
{"text1": "12 + 35", "text2": "35 + 12", "description": "顺序不同",  
"standard_similarity": 0.869}
```

输出内容 (Demo 模式):

- 测试用例描述
- 两个输入文本
- 标准相似度
- 考生计算的相似度
- 误差值
- 判定结果 (✓/✗)

评分标准:

- 每个测试用例 5 分, 共 10 组
- 允许误差: ≤ 0.03
- 通过 ≥ 8 组: 满分 50 分
- 必须满分才能进行第二部分测试

第二部分: 数据多样性 (50 分)

前提条件: 第一部分必须获得满分 (50 分)

测试内容:

- 加载 data/dataset.jsonl 数据集
- 检查数据数量、质量
- 使用考生的模型计算数据集的平均相似度

数据要求:

1. 数量要求: 1024 条
 - 每少/多 1 条扣 1 分
2. 质量要求 (每条问题扣 1 分):
 - 所有题目必须唯一 (不能有重复)

- 所有题目非空
- 每个题目最多 3 个数字
- 答案必须是 3 位数 (100-999)

3. 相似度要求:

- ≤ 0.5 : 满分 50 分
- $0.5 \sim 0.7$: 线性给分
- 0.7 : 0 分

示例数据:

JSON

```
{"problem": "818  $\ominus$  10 减 89", "answer": "719"}  
{"problem": "(98 - 30) 乘 9", "answer": "612"}  
{"problem": "28 乘 19 / 4", "answer": "133"}
```

输出内容 (Demo 模式):

- 数据数量统计
- 唯一性检查
- 数字个数检查
- 答案位数检查
- 平均相似度统计
- 扣分明细
- 最终得分

评分公式:

Plain Text

相似度得分:

- 平均相似度 ≤ 0.5 : 50 分
- 平均相似度 $0.5 \sim 0.7$: $\text{int}(50 * (0.7 - \text{avg_sim}) / 0.2)$
- 平均相似度 > 0.7 : 0 分

最终得分 = $\max(0, \text{相似度得分} - \text{数量扣分} - \text{质量扣分})$

总分构成 (100 分)

项目	分值	说明
第一部分 - 相似度计算	50 分	10 组测试用例，每组 5 分， ≥ 8 组满分
第二部分 - 数据质量	0~ -50 分	数量扣分 + 质量扣分
第二部分 - 相似度评分	0~50 分	基于平均相似度
最终得分	0~100 分	第一部分 + 第二部分

重要提示：

- 第一部分未满分 → 第二部分跳过 → 总分最多 50 分
- 第二部分得分 = 相似度得分 - 数量扣分 - 质量扣分（最低 0 分）

