Technical Writeup Overview

1. Project Overview

Project Overview

This project aims to build AI evaluation models for mathematics and essay subjects as a foundational step toward developing a lightweight, offline-compatible LLM-based learning assistant. The core objective is to support Generation Alpha children in maintaining a proper learning trajectory while interacting with AI tools.

We fine-tuned **Google's Gemma 3n E2B IT**, an instruction-tuned model released on Hugging Face, using **LoRA-based customization (including QLoRA)**. The resulting models are capable of determining correctness for math problems and assigning scores (0 to 5) for essay responses.

The project was conducted in a **hackathon setting**, with privacy and offline usability in mind. To accommodate these requirements, the model size and performance were optimized. All training data was **synthetically generated**, and LLMs were used to simulate the role of human evaluators, ensuring consistency in grading criteria.

Rather than building a simple auto-grading tool, this project explores the potential of LLMs to **analyze learners' thought processes and provide educational feedback**, paving the way for more intelligent and supportive AI learning assistants.

Fine-Tuning Stages

Stage 1: Instruction-Following Fine-Tuning (Supervised Fine-Tuning, SFT)

- **Objective**: Teach the model to follow "problem → answer" instruction formats.
- **Datasets**:
  - *Math*: Based on MathX-5M problems and generated_solution field
  - *Essay*: Based on CNN/DailyMail articles and corresponding reference summaries
- **Format**: Structured conversational prompts using <start_of_turn>system, <start_of_turn>user, <start_of_turn>model tags

---

Stage 2: Classification Fine-Tuning (Evaluation Criteria-Based)

- **Objective**: Train the model to classify how well a student's response satisfies evaluation criteria (e.g., correctness or summary quality).
- **Dataset Construction**:
  - *Math*: LLM-generated student answers vs. expected answers → Binary label (0 or 1)
  - *Essay*: LLM-generated summaries vs. reference summaries → Multiclass label (0 to 5 based on semantic similarity)

Stage 3: Pairwise Preference Fine-Tuning (Optional, Not Yet Implemented)

- **Objective**: Train the model to rank two responses based on how well they meet the evaluation criteria.
- **Status**: Planned for future work. Can be used to teach grading preferences or be combined with RLHF techniques.

2. Dataset Construction and Preprocessing

Dataset Construction

This project builds on the instruction-tuned **Gemma 3n E2B IT** model, with a three-stage dataset design aimed at enhancing learning performance. The source, structure, and preprocessing method of each dataset are as follows:

Stage 1: Instruction-Following SFT Dataset

In both math and essay domains, the goal was to train the model to generate logically coherent responses to clearly specified questions.

- **Math**: We used the “XenArcAI/MathX-5M” dataset from Hugging Face.

  Each example was constructed using the problem and generated_solution fields, and reformatted into structured conversational prompts following the format:

<start_of_turn>system

<start_of_turn>user

<start_of_turn>model

  The model was trained to follow step-by-step logical reasoning, including **thinking processes**, **intermediate calculations**, and a **final answer**, all expressed using LaTeX-formatted text.

- **Essay**: We used the cnn_dailymail dataset (version 3.0.0) from Hugging Face.

  Each article was used as the prompt, and its corresponding highlights as the target summary.

  Preprocessing involved:

  - Standardizing prompts with a template that explicitly instructed summarization.
  - Limiting input length to a maximum of 1024 tokens.

This setup trained the model to perform summarization in an instruction-following context.

Stage 2: Classification Fine-Tuning Dataset (Evaluation-Based)

Instruction-following alone is insufficient for AI to accurately **evaluate** student-generated responses. Therefore, we created additional classification datasets tailored to assessment criteria.

- **Math**:
  - We randomly sampled 10,000 examples from MathX-5M.

- Using the problem and generated_solution fields, we prompted a GPT model to generate **student-like answers**.
- Labels were assigned as follows:
    - **1** for answers matching the correct solution.
    - **0** for incorrect answers.
- We then **balanced the dataset** to include 5,000 samples with an equal number of correct and incorrect answers.
- Final format:

```
{
  "input": "Q: [Problem description] Student Answer: [Student response]",
  "label": 0 or 1
}
```

- The dataset was saved as a datasets.Dataset object in Hugging Face format for reusability.

- **Essay**:
    - We sampled 2,000 articles from the CNN/DailyMail dataset.
    - For each article, we generated a **student summary** using GPT with the prompt:

"You are a student summarizing the article below. Write a short summary."

- We then compared the **student summary** to the original **reference summary** using **Sentence-BERT** (SentenceTransformer).
- Cosine similarity scores between the two summaries were used to assign discrete labels on a 6-point scale:
    - 0.0 – 0.2 → Label 0
    - 0.2 – 0.4 → Label 1
    - 0.4 – 0.6 → Label 2
    - 0.6 – 0.75 → Label 3

        - 0.75 – 0.9 → Label 4
        - 0.9 – 1.0 → Label 5
    - Final format:

```
{
  "input": "Article: [Full article text] Student Summary: [Generated summary]",
  "label": 0 to 5
}
```

    - All datasets were saved in Hugging Face's datasets.Dataset format, with backup versions available as downloadable .zip files for use in Google Colab.

Stage 3: Pairwise Preference Dataset (Planned)

As a future step, we plan to create a pairwise dataset that compares two student responses to the same prompt, allowing the model to learn which one better aligns with the evaluation criteria. This dataset can be used for **ranking-based fine-tuning** or in **Reinforcement Learning from Human Feedback (RLHF)** setups.

3. Model Architecture and Fine-Tuning Setup

Model Architecture & Training Configuration

This project aimed to develop a lightweight AI tutor that can run **locally**, based on **Google's open-source model Gemma 3n E2B IT**. We utilized the Hugging Face Transformers library and applied **LoRA (Low-Rank Adaptation)** techniques during fine-tuning to enable efficient low-resource training.

General Specifications

- **Model**: google/gemma-3n-E2B-it (2B, instruction-tuned)
- **Fine-Tuning Method**: LoRA + QLoRA (4-bit quantization)

- **Tokenizer**: Loaded via AutoTokenizer.from_pretrained(model_id)
    - If no pad token is present, the end-of-sequence (EOS) token was used as a substitute
- **Training Framework**: Hugging Face Transformers + SFTTrainer from the trl library

---

Math Classifier Training (Binary Classification)

- **Model**: AutoModelForSequenceClassification.from_pretrained(...)
- **Number of Labels**: num_labels = 2 (correct/incorrect)
- **Input Format**:
    - Combined the *problem* and *student's answer* into the input field
    - Assigned binary label based on correctness
- **Tokenization**:

tokenizer(input, truncation=True, padding="max_length", max_length=1024)

- **Label Setup**:

  Renamed the label column to labels to ensure compatibility with the Hugging Face Trainer

- **LoRA Configuration**:

target_modules = ["q_proj", "k_proj", "v_proj", "o_proj"]

r = 8

lora_alpha = 16

lora_dropout = 0.1

- **Training Hyperparameters**:
    - batch_size = 1
    - gradient_accumulation_steps = 4
    - num_train_epochs = 2
    - learning_rate = 2e-4

- warmup_ratio = 0.03
- lr_scheduler = "constant"
- optimizer = "paged_adamw_8bit"
- Mixed precision: bf16

Essay Classifier Training (Multi-Class Classification)

- **Model**: Same Gemma 3n E2B IT base
- **Number of Labels**: num_labels = 6 (score range: 0 to 5)
- **Input Format**:
    - Combined the *article text* and *student-generated summary* into the input
    - Assigned a label based on semantic similarity score
- **Tokenization & Training Settings**:

  Identical to those used in the math classifier
- **Special Notes**:

  Labels were automatically generated using **SentenceTransformer embeddings** and **cosine similarity** between the reference and student summaries

---

Training Environment

- **Platform**: Google Colab Pro+ (A100 40GB GPU)
- **Quantization**: Applied 4-bit nf4 quantization using BitsAndBytesConfig (QLoRA)
- **Model Saving & Upload**:
    - Saved locally using .save_pretrained() to a designated Google Drive path
    - Uploaded to Hugging Face Hub via .push_to_hub()

4. Evaluation Strategy

Evaluation Strategy

The goal of this project is to train an **evaluation classifier** that can automatically assign correctness (0 or 1) or a score between 0 and 5 to student responses generated by a generative LLM. Accordingly, model performance was assessed based on **classification accuracy** and **label distribution balance**.

Math Classifier Evaluation (Binary Classification)

- **Objective**: Determine whether a student's answer to a math question is appropriate compared to the reference answer
- **Label**: 0 (Incorrect) or 1 (Correct)

Dataset Construction:

- Labels were automatically generated by computing the **cosine similarity** between the student response and the reference answer, using a predefined similarity threshold
- A **balanced dataset** was created to mitigate class imbalance, with a 1:1 ratio (e.g., 2,500 correct and 2,500 incorrect out of 5,000 total samples)

Evaluation Metrics:

- **Accuracy**
- **Confusion Matrix** (e.g., True Positive, False Positive, etc.)
- Example evaluation code will be included in future submissions

---

Essay Classifier Evaluation (Multi-Class Classification)

- **Objective**: Classify the quality of a student's summary on a 6-point scale (0 to 5)

Dataset Construction:

- Based on CNN/DailyMail articles and corresponding **student-generated summaries**

- Labels were assigned using **cosine similarity** between the student summary and the reference summary
    - Similarity scores were mapped to scores from 0 to 5 based on predefined intervals
- To reduce class imbalance, label distribution was checked across 2,000 samples, and used accordingly

Evaluation Metrics:

- **Accuracy**
- **Macro F1-Score** (to assess performance across imbalanced classes)
- **Confusion Matrix**

Validation Strategy

- No separate validation set was used during training; the **entire dataset** was used for training
- The trained model will later be evaluated in **zero-shot settings** by generating predicted labels for unseen student responses and comparing them against human- or LLM-assigned labels

5. Data Generation and Labeling Strategy

Data Generation Pipeline

Since no externally collected student response data was available for this project, we utilized **pretrained LLMs (Gemma or OpenAI GPT)** to simulate student answers and generate corresponding labels automatically. This enabled the creation of a training dataset for the evaluation classifiers. The pipeline consists of the following steps:

Step 1: Loading Problem and Reference Answer Data

- **Math Evaluation Dataset**:
    - Sampled 10,000 problems from the MathX-5M dataset on Hugging Face

- **Essay Evaluation Dataset**:
    - Loaded 2,000 news articles from the CNN/DailyMail summarization dataset

---

Step 2: Student Answer Generation

- **Math**:
    - Used an LLM to generate **plausible student answers** for each math problem
    - Prompts were designed to simulate real student responses, including **logical errors, omissions, or leaps in reasoning** to reflect realistic variance
- **Essay**:
    - For each news article, an LLM was prompted to generate a **student summary**
    - Prompts were tailored to produce summaries in **“middle-school-level English”**, ensuring variation in quality and style

---

Step 3: Automatic Labeling

- **Math**:
    - Calculated **cosine similarity** between the LLM-generated student answer and the reference answer
    - Applied a **threshold (e.g., 0.85)**:
        - Similarity ≥ threshold → label 1 (correct)
        - Similarity < threshold → label 0 (incorrect)
- **Essay**:
    - Used **Sentence-BERT** to compute **cosine similarity** between student and reference summaries

    - Mapped similarity scores to integer labels on a **6-point scale (0 to 5)** based on predefined intervals

---

Step 4: Class Balance Adjustment and Storage

- **Math**:
    - Balanced the dataset to ensure a **1:1 ratio** of correct and incorrect answers
    - Final dataset: **5,000 samples**
- **Essay**:
    - Visualized the label distribution to confirm **no severe class imbalance**
    - Used the full set of **2,000 labeled samples** without modification
- All datasets were saved in the **Hugging Face datasets.Dataset format** and backed up to **Google Drive** for reproducibility.

6. LoRA Fine-tuning Process

Fine-Tuning Strategy Overview

This project fine-tuned **Google's pretrained 2B parameter model, Gemma 3n E2B IT**, using the **LoRA (Low-Rank Adaptation)** method, adapting it for two evaluation tasks: **math scoring** and **essay grading**. The fine-tuning process was divided into three stages, as outlined below:

Step 1: Task Instruction Tuning – Math Tutor Response Generation

- **Objective**:

  To train the model to behave like a friendly and clear math tutor, generating logical explanations and step-by-step solutions in response to student-input math problems.

- **Dataset**:

    - Sampled **30,000 problems** from the [MathX-5M]() dataset.
    - Formatted prompts in the form of <system>, <user>, and <model> turns.
- **Example Format**:

<start_of_turn>system

You are GemmaMathTutor, a professional math tutor for children...

<end_of_turn>

<start_of_turn>user

What is 3x + 5 = 11?

<end_of_turn>

<start_of_turn>model

Step-by-step calculation:

3x + 5 = 11

...

Final Answer: x = 2

<end_of_turn>

- **Training**:
    - Used SFTTrainer with LoraConfig for **causal language model fine-tuning**
    - Applied **4-bit quantization** using bitsandbytes and Hugging Face Transformers

---

Step 2: Evaluation Fine-Tuning – Binary Classification (Math Answer Evaluation)

- **Objective**:

  Train a classifier to determine whether a student's answer to a math problem is correct (label 1) or incorrect (label 0)

- **Dataset**:

- Loaded a separate set of **10,000 MathX-5M samples**
- Used an LLM to generate **plausible student answers**
- Automatically labeled responses using **cosine similarity** against reference answers
- Balanced the dataset to **5,000 samples** (equal number of correct and incorrect)

- **Model**:
  - Used AutoModelForSequenceClassification
  - Set num_labels=2, applied LoRA
  - Added labels column and set dataset format to "torch"
- **Training**:
  - epochs = 2, batch_size = 1, gradient_accumulation_steps = 4
  - Model saved to **Google Drive** and uploaded to **Hugging Face Hub**

Step 3: Evaluation Fine-Tuning – Multi-Class Classification (Essay Grading)

- **Objective**:

  Train a multi-class classifier to score student-generated summaries (0 to 5) based on semantic similarity to a reference summary.

- **Dataset**:
  - Extracted **2,000 articles and reference summaries** from the **CNN/DailyMail** dataset
  - Generated **student summaries** using an LLM
  - Computed **cosine similarity** with **Sentence-BERT**, mapped to 6 score levels (0–5)
- **Model**:
  - Used AutoModelForSequenceClassification
  - Set num_labels = 6, applied LoRA
  - Converted labels to int64 for correct formatting

- **Training**:
  - Followed the same hyperparameter settings as in Step 2
  - Verified correct loss calculation and logits handling for multi-class task
  - Model saved and uploaded to **Hugging Face Hub**

7. Training Environment & Runtime Efficiency

Runtime Environment & Optimization Strategy

This project was executed in the **Google Colab Pro+** environment, primarily utilizing **NVIDIA A100 (40GB)** GPU instances. Some parts of the data generation pipeline—particularly those involving the **OpenAI API**—were conducted using CPU resources.

Environment Configuration

- **Platform**: Google Colab Pro+
- **GPU**: NVIDIA A100 40GB
- **Python Version**: 3.11

Key Libraries Used:

- transformers
- datasets
- peft
- trl
- bitsandbytes
- sentence-transformers
- scikit-learn

Model-Specific Configuration

Both the math and essay fine-tuning tasks were performed using **4-bit QLoRA quantization** to maximize memory efficiency.

BitsAndBytesConfig Parameters:

load_in_4bit = True

bnb_4bit_quant_type = "nf4"

bnb_4bit_compute_dtype = torch.bfloat16

bnb_4bit_use_double_quant = True

LoRA Configuration:

r = 8

lora_alpha = 16

lora_dropout = 0.05

target_modules = ["q_proj", "v_proj"]

Training Time (Estimates on A100 GPU)

- **Math Binary Classification Model**
  - Dataset: 5,000 samples
  - Epochs: 2
  - Training Time: **~35–45 minutes**
- **Essay Multi-Class Classification Model**
  - Dataset: 2,000 samples
  - Epochs: 2
  - Training Time: **~20–25 minutes**

LLM-Based Data Generation

- **Model**: OpenAI GPT-3.5
- **Estimated Time**:
  - Over **1 hour per 1,000 samples**
- **Note**: To manage **API cost and latency**, data generation was capped at **approximately 2,000 examples**

Optimization & Bottleneck Handling

- During training, intermediate results were **regularly saved to Google Drive** to prevent loss in case of session termination.
- A temporary directory (/content) was used to mitigate accidental data loss.
- In case of **API cost limitations**, alternative **asynchronous or CPU-based fallback logic** (e.g., cosine similarity scoring without external API calls) was considered for scalability.

8. Results & Model Saving/Sharing

Model Saving & Deployment

All fine-tuned models developed in this project were saved both **locally** and on the **Hugging Face Hub** for version control and reproducibility. Each model corresponding to an experiment is organized as follows:

Math Binary Classification Model

- **Local Save Path**:

  /content/drive/MyDrive/gemma_finetuned_math_classifier

- **Hugging Face Hub URL**:

  https://huggingface.co/LeannaJ/gemma3n-math-eval

Essay Multi-Class Classification Model

- **Local Save Path**:

  /content/drive/MyDrive/essay_eval_multiclass

- **Hugging Face Hub URL**:

  https://huggingface.co/LeannaJ/essay_evaluation

Saving Workflow

After training, both the model and tokenizer were saved using the following code:

```
save_path = "/content/drive/MyDrive/your_model_directory"
model.save_pretrained(save_path)
tokenizer.save_pretrained(save_path)

repo_id = "LeannaJ/your_model_repo"
model.push_to_hub(repo_id)
tokenizer.push_to_hub(repo_id)
```

Considerations for Hugging Face Upload

- **Authentication**: Use your Hugging Face token within Colab or manually authenticate via the huggingface-cli.
- **Repository Visibility**: The public/private setting can be managed directly on the Hugging Face repository page.
- **Reusability**: Once uploaded, the models can be **loaded and evaluated directly on platforms like Kaggle**.

9. Limitations & Future Directions

Limitations

Model Architecture Constraints:

- The google/gemma-3n-E2B-it model is optimized for **causal language modeling**, not for sequence classification tasks.
- To adapt it for classification, we had to apply AutoModelForSequenceClassification manually, which required **custom handling** due to configuration compatibility issues.

Subjectivity in Evaluation Criteria:

- Especially in essay grading, the labels generated via **LLM-based summarization and cosine similarity** may not fully align with those of **human educators**.
- To ensure reliable automatic evaluation, further **rule-based refinement** or **feedback from human graders** may be necessary.

Limited Training Data Volume:

- Due to time and resource constraints, the training was limited to **5,000 math samples** and **2,000 essay samples**.
- This amount is insufficient to capture the **diversity of question types and difficulty levels**, which may limit the model's ability to generalize.

Future Work

Prompt Engineering and Instruction Fine-Tuning:

- We plan to improve the **prompt design** used for answer validation and summarization, allowing the evaluation LLM to generate **more accurate and context-aware feedback**.
- Custom prompt formats will be explored for both math and essay domains, tailored to handle diverse input types and student response styles.

Granular Evaluation Criteria:

- For essays, we aim to go beyond a single 0–5 score by introducing **rubric-based evaluations**, such as:
    - Logic and coherence
    - Grammar and syntax
    - Topic relevance
- For math, future versions may include evaluation on:
    - **Reasonableness of problem-solving approach**
    - **Accuracy of calculation steps**, in addition to final answer correctness

Model Adaptation Experiments (within Gemma family):

- Although we utilized the Gemma causal model with classification wrappers, we plan to explore **other configurations or architectural variants** within the Gemma family, if made available, to better suit evaluation tasks.

Offline & Privacy-Conscious Deployment:

- While current training and inference rely on **cloud-based APIs** and **Hugging Face Hub**, future work will focus on making the models **fully offline-compatible**.
- This includes converting the models to **GGUF format** and testing inference performance on **mobile or edge devices**, enhancing accessibility and preserving privacy.