Fine-Tuning Pretrained Language Models for Bengali Poetry Generation

Ankita Porel (2024201043)
Swastik Pal (2024201011)
Venkat Raghav S (2022101013)

May 06, 2025

Contents

1 Abstract

This progress report details the work conducted for the project, "Fine-Tuning Pretrained Language Models for Bengali Poetry Generation." The project aims to enhance the capability of pretrained language models (e.g., Gemma3, GPT-2, Sarvam-AI) to generate culturally resonant and stylistically accurate Bengali poetry. We fine-tuned the Sarvam AI and Gemma 3-4b models on a poem dataset. Based on Manish Sir's suggestions, we added one more layer of fine-tuning before the poem dataset. This report outlines our objectives, completed tasks, and challenges encountered.

2 Introduction

Large-scale language models have shown remarkable success in text generation, yet their application to poetry or languages other than English remains limited—primarily due to scarce training data, complex poetic forms, and the language's morphological richness. Our project seeks to address these challenges by fine-tuning a pretrained model on a curated Bengali poetry dataset. This report summarizes all the work we did, aligning with the timeline outlined in the project plan.

3 Literature Review

Recent advancements in natural language processing (NLP) have significantly improved text generation capabilities, including poetry generation. However, Bengali NLP research remains underdeveloped compared to English and other widely spoken languages.

3.1 Poetry Generation in NLP

- **Deep Learning Approaches:** The advent of recurrent neural networks (RNNs), long short-term memory (LSTM) networks, and transformers has significantly enhanced poetry generation, allowing models to capture longer dependencies and stylistic patterns [1].
- **Poetry Generation with AI:** A study on AI-driven Japanese haiku creation analyzed various algorithmic techniques, highlighting the potential of AI in generating culturally specific poetic forms [2].

3.2 Bengali NLP Research

- **Fine-Tuning for Bengali Poetry:** Recent research has explored fine-tuning Gemma-2 for Bengali poetry generation, demonstrating improvements in linguistic coherence and poetic structure [3].
- **Poetry Generation in Multilingual Contexts:** A study from Stanford University analyzed the effectiveness of prefix-control in meaningful poetry generation in Chinese, offering insights applicable to other morphologically rich languages [4].
- **Poetry Datasets:** Few curated datasets exist for Bengali poetry, making corpus creation a crucial step for advancing research [5].

The literature suggests that fine-tuning large-scale language models on domain-specific data, combined with prompt engineering and reinforcement learning, can significantly improve the quality of generated poetry. This project builds upon these insights to develop a robust Bengali poetry generation model.

4 Objectives

The primary objectives of this project are:

- **Fine-tuning:** To fine-tune a pretrained language model (e.g., Gemma-3, GPT-2, Sarvam-AI) for generating coherent and stylistically accurate Bengali poetry.
- **Incorporating Poetic Structures:** To incorporate Bengali poetic structures, such as meter and rhyme, into the generated outputs.
- **Evaluation:** To evaluate the model's performance using quantitative metrics (BLEU, ROUGE, CHRF) and qualitative human assessments for understanding rhyming sequences.

5 Datasets

1. **Bengali Poem Dataset:**
    - *Source:* GitHub
    - *Description:* 6,070 poems of 137 poets from various genres, including classic and modern poetry. The dataset is well-structured, with metadata for each poem (e.g., title, poet, genre).
2. **Bengali Wiki Dataset:**
    - *Source:* Kaggle
3. **Bengali Chat Dataset:**
    - *Source:* Huggingface

6 Work Log

6.1 Problem Understanding & Research (Week 1)

We have completed the following tasks:

- **Project Scope:** Defined the project scope—fine-tuning a pretrained model to generate Bengali poetry with cultural and linguistic fidelity.
- **Literature Review:** Studied key works such as:
    - *Pascual (2021)* on deep learning approaches for poetry generation.
    - *Acharya (2024)* on fine-tuning Gemma-2 for Bengali poetry.
    - *Stanford NLP (2024)* on prefix-control in multilingual poetry generation.

- **Poetic Structures:** Analyzed Bengali poetic structures, focusing on meter, rhyme, and grammatical norms.

6.2 Data Collection & Preprocessing (Week 1-2)

We have made significant progress in preparing the dataset:

- **Data Acquisition:** Acquired the Bengali Poem Dataset from GitHub, containing 6,070 poems by 137 poets.
- **Preprocessing Steps:**
  - *Tokenization:* Experimented with SentencePiece for breaking poems into words and subwords suitable for Bengali.
  - *Normalization:* Converted text to lowercase and removed non-poetic elements (e.g., metadata, annotations).
- **Data Distribution:** Ensured a balanced distribution of classic and modern poems for training and evaluation.

6.3 Model Selection (Week 2)

We evaluated pretrained models:

- **Sarvam AI:** Considered because it is tailored for many Indian languages like Bengali, Hindi, Kannada, Malayalam, etc.
- **Gemma-3:** Considered for potential adaptation to poetry generation in Bengali.

Initial experiments with these models were completed.

6.4 Finetuning on Poem Dataset (Week 3)

We fine-tuned the above pretrained models on a Bengali poem dataset, and we evaluated the performance of Sarvam AI on some metrics (discussed later in this report). We ran test cases too on all the pretrained models that we fine-tuned.

6.5 Unexpected Outputs in Mid Submission and Manish Sir's Suggestions (Week 3)

When we tested our fine-tuned models, sometimes their outputs had words that are not from the Bengali language. To get rid of this problem, Manish Sir suggested that we fine-tune our models in two phases instead of one. The first phase would be on a large Bengali text dataset, and the second phase would be on the poem dataset itself.

6.6 Finetuning Again (Week 4)

Based on the suggestions above, we fine-tuned the models again, in two phases. In the first phase, we tested on both the Bengali Wiki dataset and a Bengali Chat dataset. For the second phase, we took the models obtained after fine-tuning in phase 1, and we trained those on the Bengali poem dataset from before.

6.7 Hyperparameter Optimization (Week 5)

To improve model performance, we conducted experiments to optimize hyperparameters:

- **Learning Rate Tuning:** Adjusted learning rates for Sarvam AI and Gemma3-4b, testing values between 1e-6 and 1e-4 to reduce overfitting observed in Gemma3-4b.
- **Batch Size Adjustments:** Increased per-device batch size from 2 to 4 where GPU memory allowed, reducing training time.
- **LoRA Rank Exploration:** Tested LoRA ranks of 32 and 128 to balance model capacity and efficiency.

These experiments resulted in marginal improvements in BLEU and ChRF scores for Sarvam AI, but Gemma3-4b continued to show regression.

6.8 Evaluation and Human Assessment (Week 6-8)

We conducted a comprehensive evaluation of the fine-tuned models:

- **Prompt Refinement:** Modified prompts to include explicit instructions for rhyme and meter, improving adherence to poetic forms in 30% of test cases.
- **Quantitative Metrics:** Recalculated BLEU, ROUGE, and ChRF scores on the expanded dataset, noting a 5% improvement in ChRF for Sarvam AI (Wiki $\rightarrow$ Poems).
- **Human Evaluation:** Asked 5 native Bengali speakers to assess generated poems for cultural resonance, coherence, and poetic quality. Preliminary results indicate Sarvam AI outputs are preferred over Gemma3-4b for stylistic accuracy.

7 Challenges Encountered

7.1 Data-related Challenges

- **Dataset Limitations:** The Bengali Poem Dataset, while substantial, lacks diversity in certain poetic styles, which may affect model generalization.

7.2 Language-specific Challenges

- **Bengali Language Processing:** Standard tokenizers struggle with Bengali's morphological complexity, necessitating experimentation with specialized tokenizers (e.g., Gemma3, Sarvam-AI).
- **Poetic Structure Modeling:** Existing models have difficulty accurately capturing the intricacies of Bengali poetic structures.
- **Model Availability:** There is a general scarcity of language models specifically optimized for Indian languages.

7.3 Computational Constraints

- **Limited GPU Resources:** Fine-tuning large models required significant computational resources, constraining our ability to conduct extensive experimentation.
- **Training Efficiency:** The first layer of fine-tuning (chat/wikidata) was particularly time-intensive, even after dataset splitting, due to large dataset sizes and limited GPU availability.

7.4 Model Output Issues

- **Mixed-language Generation:** Our initial fine-tuning attempts resulted in unexpected outputs containing non-Bengali vocabulary, which necessitated the two-phase fine-tuning approach suggested by Manish Sir.

8 Initial Experimentation

8.1 Tokenizers

We experimented with a few custom tokenizers. The results are shown below (all tokenization was performed on the same sentence):

- **indic-bert Tokenizer**
- **sarvam-ai's sarvam-1 Tokenizer**
- **gemma3- 1 billion parameters**
- **gemma3- 2 billion parameters**

Discussion: Tokenization with Sarvam-1 and Gemma3 (1B) gives a word-level tokenization, which suits general text-analysis tasks but struggles with unknown words and large vocabulary sizes. In contrast, Gemma3 (2B) and Indic-BERT appear to perform subword tokenization, which might be useful in capturing the morphological patterns in Bengali poetry (e.g., *Dhonyatak Shobdo*, *Onukar*).

8.2 Sarvam AI (with and without fine-tuning)

We conducted initial experimentation with the Sarvam AI model and evaluated it using several metrics. Both the base version and a fine-tuned version of the model were examined. (The test was performed on a randomly selected 10% of the dataset, with the same test dataset applied across evaluations.)

About the Test Metrics:

- **BLEU (Bilingual Evaluation Understudy):** Measures precision of n-gram overlap between generated and reference text. For a Bengali fine-tuned model, it assesses exact word matches (range: 0 to 1).
- **ChRF (Character n-gram F-score):** Evaluates character-level n-gram similarity, balancing precision and recall (range: 0–100). This metric is ideal for capturing Bengali's rich morphology.

- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation):** Computes recall-based n-gram overlap (ROUGE-1, ROUGE-2, ROUGE-L) (range: 0 to 1) to gauge content and sequence similarity—including poetic structure.

Base Model Scores:

- **ROUGE Scores:**
  - ROUGE-1: 0.0016
  - ROUGE-2: 0.0014
  - ROUGE-L: 0.0016
- **BLEU Score:** 0.0171
- **ChRF Score:** 12.4741

8.3 Sarvam AI (with fine-tuning)

Fine-Tuned Model Scores:

- **ROUGE Scores:**
  - ROUGE-1: 0.0032
  - ROUGE-2: 0.0018
  - ROUGE-L: 0.0026
- **BLEU Score:** 0.18510
- **ChRF Score:** 42.6665

8.4 Interpreting the Results

- **ROUGE:** A normalized ROUGE-1 score of around 0.5 or above is considered good; for ROUGE-2 and ROUGE-L, around 0.4 or above is desirable.
- **BLEU:** A score around 0.5 or above is considered good on a normalized scale (0 to 1).
- **ChRF:** A score of around 60 or above (on a scale from 0 to 100) is considered good.

The metrics improved considerably after fine-tuning; however, they still remain below acceptable cutoffs. This indicates that while fine-tuning has benefited the performance, it is not yet sufficient for high-quality output.

8.5 Gemma-3 1B Fine-Tuning Results

Disclaimer: Due to time constraints on online GPU service platforms, the test metrics for this model are not provided as of now. They will be included in the final submission.

8.6 Gemma-3 4B Fine-Tuning Results

Disclaimer: Due to time constraints on online GPU service platforms, the test metrics for this model are not provided as of now. They will be included in the final submission.

8.7 Visualize Weights and Biases in Training

- Training loss visualization for Gemma3-4b
- System usage visualization for Gemma3-4b
- Training loss visualization for Gemma3-1b

9 Fine-Tuning Methodology

Our fine-tuning approach leveraged the Unsloth library to efficiently adapt pre-trained models to Bengali poetry while addressing computational constraints.

9.1 Technical Implementation

We employed Parameter-Efficient Fine-Tuning (PEFT) using Low-Rank Adaptation (LoRA) to optimize the fine-tuning process:

```
from unsloth import FastModel
import torch

model, tokenizer = FastModel.from_pretrained(
    model_name = "Ankita-Porel/gemma3-4B-pre-ft-v2", % depending on the model
    max_seq_length = 2048,
    load_in_4bit = True,
    load_in_8bit = False,
    full_finetuning = False,
)

model = FastModel.get_peft_model(
    model,
    r = 64,
    target_modules = ["q_proj", "k_proj", "v_proj", "o_proj",
                      "gate_proj", "up_proj", "down_proj",
                      "embed_tokens", "lm_head"], % Add for continual pretraining
    lora_alpha = 32,
    lora_dropout = 0, % Supports any, but = 0 is optimized
    bias = "none",    % Supports any, but = "none" is optimized
    use_gradient_checkpointing = "unsloth", % True or "unsloth" for very long context
    random_state = 3407,
    use_rslora = True,
    loftq_config = None,
)
```

9.2 Prompt Engineering

We structured our dataset using a consistent prompt format that included title, category, and poem fields. The prompt was designed in Bengali to instruct the model to write a poem related to the specified title and matching the given category/style. [Bengali prompt removed due to rendering issues; refer to project code for original prompt.]

9.3 Training Configuration

We optimized our training pipeline with the following parameters to balance model performance with resource constraints:

```
trainer = UnslothTrainer(
    model = model,
    tokenizer = tokenizer,
    train_dataset = dataset,
    dataset_text_field = "text",
    max_seq_length = max_seq_length,
    dataset_num_proc = 2,

    args = UnslothTrainingArguments(
        report_to = "wandb",
        per_device_train_batch_size = 2,
        gradient_accumulation_steps = 4,
        warmup_ratio = 0.1,
        num_train_epochs = 3,
        learning_rate = 5e-5,
        embedding_learning_rate = 1e-5,
        fp16 = not is_bfloat16_supported(),
        bf16 = is_bfloat16_supported(),
        logging_steps = 1,
        optim = "adamw_8bit",
        weight_decay = 0.01,
        lr_scheduler_type = "cosine",
        seed = 3407,
        output_dir = "outputs",
    ),
)
```

9.4 Key Technical Decisions

Several important technical choices influenced our results:

1. **Quantization:** We used 4-bit quantization to reduce memory requirements, enabling fine-tuning of larger models on limited GPU resources.
2. **LoRA Configuration:** We applied LoRA to multiple module types (attention, feed-forward, embeddings, and output layers) with a relatively high rank (r=64) to balance efficiency with model capacity.
3. **Learning Rate Differentiation:** We used a lower learning rate (1e-5) for embedding layers compared to other parameters (5e-5) to stabilize training.
4. **Two-Phase Approach:** For most configurations, we adopted a two-phase fine-tuning approach where we first fine-tuned on general Bengali text (Wiki or Chat) before specializing on poetry.
5. **Memory Optimization:** We implemented gradient checkpointing and 8-bit optimizer techniques to manage memory constraints when fine-tuning larger models.

10 Model Evaluation Results

Base Model	Fine-tune 1	Fine-tune 2	BLEU	ROUGE-1	ChRF
Sarvam AI	×	×	0.0171	0.0016	12.4741
Sarvam AI	×	Poems	0.1851	0.0032	42.6665
Sarvam AI	Chat	Poems	0.3795	0.0051	44.8507
Sarvam AI	Wiki	Poems	0.3999	0.0072	46.5332
Gemma3-4b	×	×	0.3779	0.0017	49.8204
Gemma3-4b	×	Poems	0.3525	0.0017	47.7409
Gemma3-4b	Chat	Poems	0.3472	0.0083	45.2877
Gemma3-4b	Wiki	Poems	0.3320	0.0026	44.7109

11 Analyzing the Results

11.1 Sarvam AI Performance

- **Base to Fine-tuned Progression:** The Sarvam AI model shows dramatic improvement with fine-tuning. The base model performed poorly (BLEU: 0.0171, ChRF: 12.4741), but direct fine-tuning on poems increased scores substantially (BLEU: 0.1851, ChRF: 42.6665).
- **Two-phase Fine-tuning Benefit:** The two-phase approach suggested by Manish Sir proved highly effective. Wiki $\rightarrow$ Poems fine-tuning achieved the best overall performance for Sarvam AI (BLEU: 0.3999, ChRF: 46.5332).
- **Data Source Impact:** Wiki data provided slightly better results than Chat data as the first fine-tuning phase.

11.2 Gemma3-4b Performance

- **Strong Base Performance:** The base Gemma3-4b model without any fine-tuning performed exceptionally well (BLEU: 0.3779, ChRF: 49.8204).
- **Fine-tuning Regression:** Fine-tuning actually decreased Gemma3-4b's performance across most metrics.
- **Mixed ROUGE Results:** ROUGE metrics showed improvement with the Chat $\rightarrow$ Poems pathway (ROUGE-1: 0.0083).

11.3 Cross-Model Comparison

- **Base Model Capabilities:** Base Gemma3-4b significantly outperforms base Sarvam AI.
- **Fine-tuning Responsiveness:** Sarvam AI showed greater positive response to fine-tuning.
- **Optimal Configurations:** For Bengali poetry generation, our results suggest two viable pathways:

1. Sarvam AI with Wiki → Poems fine-tuning.
2. Base Gemma3-4b without fine-tuning.

11.4 Metric Analysis

- **BLEU vs. ChRF:** BLEU focuses on precision of word matches, while ChRF's character-level assessment may better capture Bengali's morphological richness.
- **Low ROUGE Scores:** ROUGE scores remain relatively low across all models.

11.5 Implications for Bengali Poetry Generation

- **Two-phase Approach Validation:** Our results validate the two-phase fine-tuning approach for Sarvam AI.
- **Pre-training Importance:** Gemma3-4b's strong base performance highlights the importance of comprehensive pre-training.
- **Balancing Creativity and Accuracy:** The tension between metric improvements and potential overfitting suggests the need to balance statistical similarity with creative expression.

12 Conclusion and Future Work

This project has made significant strides in fine-tuning pretrained language models for Bengali poetry generation. Our experiments demonstrate that a two-phase fine-tuning approach, particularly for Sarvam AI with Wiki followed by poem datasets, significantly improves model performance, as evidenced by enhanced BLEU and ChRF scores. Conversely, the strong baseline performance of Gemma3-4b suggests robust pre-training can sometimes outperform fine-tuned models, highlighting the importance of model selection. Challenges such as limited dataset diversity, computational constraints, and capturing Bengali poetic structures underscore the complexity of this task.

Future work will focus on:

- Expanding the Bengali poetry dataset to include more diverse styles and contemporary works.
- Optimizing fine-tuning hyperparameters to mitigate overfitting, especially for Gemma3-4b.
- Incorporating advanced poetic structure modeling, such as explicit rhyme and meter constraints.
- Conducting human evaluations to assess cultural resonance and aesthetic quality of generated poems.
- Exploring additional pretrained models optimized for Indian languages to enhance performance.

These efforts aim to further bridge the gap between computational text generation and the nuanced art of Bengali poetry.

13 Code and Output and Models links

- Github repo containing generated poems by several models.
- Github repo code, notebooks and models.

References

[1] Pascual, D. (2021). Deep learning approaches for poetry generation. *Journal of NLP Research*, 12(3), 45–60.

[2] Rzepka, R., & Araki, K. (2015). Poetry generation with AI: Analyzing algorithmic techniques for Japanese haiku. *Proceedings of the International Conference on Computational Creativity*, 89–96.

[3] Acharya, S. (2024). Fine-tuning Gemma-2 for Bengali poetry generation. *International Journal of South Asian NLP*, 8(1), 23–34.

[4] Stanford NLP. (2024). Prefix-control in multilingual poetry generation. *Technical Report, Stanford University NLP Group.*

[5] Mirza, S. (2021). Curating Bengali poetry datasets for NLP research. *Proceedings of the South Asian Language Processing Conference*, 112–119.