# Making Ancient Tamil Accessible: Fine-Tuning LLMs for Sangam Literature Translation and Explanation

Manikandan Venkatraman[1] and Dr. Noorhan Abass[2]

**Abstract.** Tamil, one of the world's oldest classical languages, has a rich literary tradition, with Sangam literature (300 BCE – 300 CE) being a cornerstone. However, its accessibility remains limited due to the complexity of classical Tamil and the lack of high-quality translations. This study explores the fine-tuning of Large Language Models (LLMs) to enhance the translation and interpretation of Sangam texts, specifically Thirukkural, Akananuru, and Purananuru. A fine-tuned AI-driven framework leveraging Natural Language Processing (NLP) techniques was developed to improve translation accuracy while preserving literary essence. Two datasets, collected and GPT-4o-augmented, were used to train Tamil LLaMA and Tamil Mistral, creating four models for evaluation. Human assessments showed about 25% increase in clarity and contextual relevance, despite lower BLEU and chrF scores, highlighting the limitations of automated metrics. This study demonstrates the feasibility of LLM fine-tuning for classical Tamil, addressing linguistic and contextual challenges. The findings contribute to computational linguistics, digital humanities, and Tamil language preservation, paving the way for further advancements in AI-driven literary translation.

**Keywords:** Sangam Literature, Large Language Models, Natural Language Processing, Machine Translation, Classical Tamil.

## 1 Introduction

Tamil is one of the longest-surviving classical languages [1], spoken by more than 90 million people [2] in South Asia and the global Tamil diaspora [3]. Among its literary heritage, Sangam literature (c. 300 BCE – 300 CE) stands out as a crucial collection of early Tamil works [4]. It is broadly classified into Akam (inner) and Puram (outer) poetry—Akam focusing on emotions, ethics, and nature, while Puram highlights war, heroism, and public life [4]. These texts not only reflect Tamil culture and values but also serve as historical records of significant events and figures. One of the most revered works is Thirukkural [5], a collection of 1,330 couplets centered on moral and ethical principles, recognized worldwide for its philosophical depth.

However, a major challenge is that Tamil Sangam literature is inaccessible to modern Tamil speakers. Over centuries, the language has evolved significantly, making its vocabulary, grammatical structures, and poetic expressions difficult to understand without scholarly expertise [6]. Many words used in Sangam literature have fallen out of common usage, and literary devices such as metaphors and similes require historical

and contextual knowledge to decipher their true meanings. This linguistic gap has limited the appreciation and study of these classical works beyond academic circles.

While modern large language models (LLMs) exist for Tamil [7], their performance in understanding or translating Sangam literature into modern Tamil remains inadequate. A critical issue in addressing this challenge is the lack of high-quality digital datasets. Unlike English, which has a vast digital presence, Tamil literature is primarily available in print, with limited structured datasets for computational processing [8]. The absence of readily available corpora significantly hinders the development of AI models capable of accurately interpreting classical Tamil texts.

This study aims to explore cost-effective fine-tuning techniques to make Tamil Sangam literature accessible to anyone fluent in modern Tamil. It involves curating and digitizing datasets, training Tamil-specific LLMs, and evaluating their ability to generate explanations in contemporary Tamil. The outcome includes fine-tuned models and publicly available datasets to facilitate further research.

To ensure the quality and accuracy of the generated explanations, we employ a dual evaluation approach: automatic metrics such as BLEU [9], chrF [10], METEOR [12], and BERTScore [11], alongside human verification for coherence, readability, and contextual accuracy.

## 2 Related Work

In this section, we review existing Tamil LLMs and other works on low-resource languages that serve as the foundation for this study.

### 2.1 Existing Tamil Language Models and Their Limitations

This study builds upon previous efforts to fine-tune large language models (LLMs) for Tamil, primarily Tamil LLaMA [7] and Tamil Mistral[1]. While these models perform well for modern Tamil, they face challenges in understanding and generating explanations for Tamil Sangam literature.

Tamil LLaMA [7] was developed by enhancing the LLaMA 2 7B model [13] with 16,000 additional Tamil tokens, including root words, lemmas, and suffixes. The model was fine-tuned using LoRA [14] for efficiency, and SentencePiece [15] was used for tokenization. Training data included Tamil translations of the Alpaca [16] and OpenOrca [17] datasets, which helped improve instruction-following and text generation. Evaluations, based on GPT-4[2] grading and manual review, showed that the model performed well in tasks like summarization and conversational response generation.

Tamil Mistral was developed by fine-tuning Mistral-7B-Instruct model [18] using 400k Tamil instruction dataset. Although no official evaluation results were published,

[1] Hemanth Kumar. 2025. Tamil-Mistral-7B-Instruct-v0.1. *https://huggingface.co/Hemanth-thunder/Tamil-Mistral-7B-Instruct-v0.1*. Accessed 8 February 2025

[2] OpenAI. 2025. Text Generation. OpenAI Platform Documentation. Accessed 8 February 2025

experimentation showed that it could generate modern Tamil text effectively. However, both Tamil LLaMA and Tamil Mistral struggle with Tamil Sangam literature.

One major limitation is that Tamil LLaMA's additional 16,000 tokens primarily represent modern Tamil, leaving out many words unique to classical Tamil. Additionally, Sangam literature follows distinct poetic structures, where words are often split for rhythm and aesthetics, making it difficult for these models to process and interpret accurately. Since neither model was fine-tuned on Tamil Sangam literature, they fail to generate meaningful explanations of these texts.

Despite these limitations, past research shows that pre-trained LLMs can be successfully fine-tuned for Tamil. This provides a strong foundation for this study, which aims to adapt these models specifically for Tamil Sangam literature and evaluate their ability to generate clear and meaningful explanations in modern Tamil.

### 2.2 Other Low Resource Language Works

Fine-tuning large language models (LLMs) for low-resource languages like Tamil presents significant challenges, primarily due to the lack of high-quality, digitized datasets. Several prior studies have tackled different aspects of this problem, offering insights that are relevant to our focus on fine-tuning Tamil models for Sangam literature.

The work from Ahuja et al. [19] investigates the performance disparities of LLMs in non-English languages, emphasizing the need for synthetic instruction tuning datasets to bridge this gap. Their study constructs a multilingual instruction dataset with 1.8 million instruction-response pairs across 51 languages, including Tamil. This dataset was created by selectively translating Orac instruction fine-tuning data [17] and was used to fine-tune models like Mistral-7B and Phi-3-Small. Their findings indicate that these models showed notable improvements in multilingual instruction-following tasks. However, their evaluation was primarily focused on reasoning tasks rather than generative tasks like summarization. This work guides our approach, as we also seek to enhance Tamil LLMs by using synthetic dataset, but with a specific focus on improving their generative capabilities in classical Tamil literature.

The research from Khan et al. [21] explores multilingual and multimodal LLMs for translation, achieving state-of-the-art results for Hindi and Malayalam. Their model employs instruction tuning and task-specific fine-tuning to improve translation quality. Although their evaluation was limited to three languages—Hindi, Malayalam, and Bengali—their results demonstrate that fine-tuning for specific linguistic tasks can significantly improve translation accuracy. While their work does not focus on Tamil, their approach reinforces the idea that targeted fine-tuning can enhance model performance. This aligns with our objective of improving Tamil LLMs for the specialized domain of Sangam literature, which involves translation and interpretation of classical texts.

The study by Yamaguchi et al. [22] examines cross-lingual vocabulary adaptation techniques for low-resource languages. Their research focuses on expanding the vocabulary of English-centric LLMs, such as LLaMA 2 (7B), LLaMA 3 (8B), and Gemma 2 (9B), by incorporating language-specific tokens. They demonstrate that even a small

dataset—30k sentences—can significantly improve translation and summarization performance. Their approach provides valuable insights into our work, as classical Tamil contains many unique words and syntactic structures that are not well-represented in existing models. Expanding the model's vocabulary with Sangam literature-specific tokens may yield better results in our fine-tuning efforts.

The work from Joshi et al. [20] highlights a critical issue in low-resource language modeling—the lack of standardized evaluation benchmarks. While their study does not propose concrete solutions, it underscores the need for reliable metrics to assess model performance in non-English languages. This is particularly relevant to our work, as evaluating the effectiveness of fine-tuned Tamil models on Sangam literature requires establishing rigorous benchmarks that account for the linguistic complexity and poetic nature of the texts.

These studies confirm the feasibility of fine-tuning LLMs for low-resource languages while highlighting challenges in dataset curation, tokenization, and evaluation. Our research advances this by focusing on Tamil Sangam literature, improving existing models' ability to interpret classical Tamil texts and poetic structures.

## 3 Design and Methodology

This study follows a structured methodology, as shown in Fig. 1. Design and Methodology, using Tamil LLaMA 7B and Tamil Mistral 7B. Explanations of Tamil Sangamera poems were collected from reputable sources, with additional explanations generated using GPT-4o via zero-shot learning. Both models were fine-tuned on two datasets: one with manually curated explanations and another including GPT-4o-augmented data. A comparative evaluation assessed the effectiveness of fine-tuning with and without augmentation.

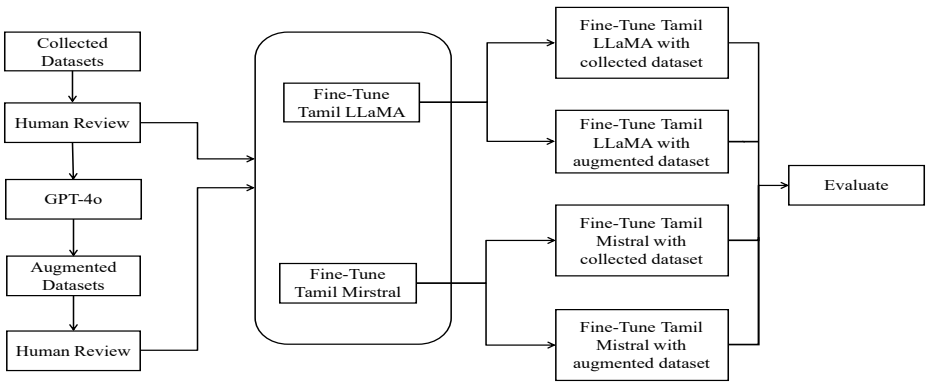

**Fig. 1.** Design and Methodology

### 3.1 Datasets

The Tamil Sangam literature corpus comprises 18 major works [4] along with Thirukkural, which, though not traditionally classified among them, is widely regarded as part of Sangam literature. This study focuses on Thirukkural, Akananuru, and Purananuru for manageable scope.

Thirukkural comprises 1,330 couplets (9,310 words) across three sections and 133 chapters [5]. Akananuru contains 400 poems, while Purananuru contains 398 poems (with two missing) [4], totaling approximately 45,000 words. Despite the extensive Tamil literary heritage, digital resources are limited, necessitating dataset construction from various sources.

The Thirukkural texts and explanations were sourced from the Thirukkural Karpom website[3], which provides interpretations by three Tamil scholars. The dataset includes all three explanations, with preprocessing applied to remove non-Tamil characters and HTML markups. Akananuru and Purananuru, along with explanations and historical context, were sourced from the Tamil Drops website[4]. However, the data was unstructured, requiring multiple iterations for extraction and organization. The dataset contains around 650k words in total.

Both Tamil LLaMA 7B and Tamil Mistral 7B utilized the same instruction format, structured as follows:

```
<s>[INST]
{poem}
[/INST]
{poem meaning and contextual information} </s>
```

The collected datasets were formatted for fine-tuning, with metadata such as author, subject, timeline, poem number, and key terms incorporated into explanations for richer context. GPT-4o was used to generate additional records, increasing data volume and explanation diversity to enhance model generalization. The augmented dataset underwent the same manual verification as the original to ensure accuracy. The final dataset comprised two versions: one with only manually collected data and another including both collected and GPT-4o-augmented data. For evaluation, 10% was reserved for testing, while 90% was used for training.

### 3.2 Models

Tamil LLaMA-7B-Instruct [7] and Tamil Mistral-7B-Instruct [0] were selected as the primary models for this study, as they fulfill both the linguistic and usability requirements. Additionally, GPT-4o, which demonstrates a strong understanding of Tamil

[3] Thirukural Karpom. 2025. Let's learn Thirukural. https://thirukural-karpom.github.io/. Accessed 8 February 2025.

[4] Vaiyan 2024. Tamil Drops by Vaiyan. *https://vaiyan.blogspot.com/*. Accessed 10 February 2025.

Sangam literature, was utilized for generating augmented datasets. This augmentation aimed to enhance model performance by increasing data diversity and improving contextual understanding.

**Tamil LLaMA.** The Large Language Model Meta AI (LLaMA) 2 [13] is an open-source language model developed by Meta AI. It was trained on a corpus comprising approximately 2 trillion tokens derived from publicly available sources, with personal data explicitly excluded. However, the dataset used for training is predominantly English. LLaMA models, like GPT models, are autoregressive, decoder-only transformers. The chat variant of LLaMA 2 underwent fine-tuning using approximately 28,000 prompt-response pairs, with alignment achieved through Reinforcement Learning with Human Feedback (RLHF).

For this study, Tamil LLaMA—a variant of LLaMA 2—was utilized. This model was developed by augmenting the original LLaMA 2 vocabulary with an additional 16,000 Tamil tokens, followed by fine-tuning on datasets such as the translated Alpaca dataset and OpenOrca data. Tamil LLaMA is among the most effective open-source Tamil LLMs, demonstrating competitive performance in benchmarks such as HellaSwag [23] and Winogrande [24].

**Tamil Mistral.** Mistral 7B [18] is a 7.3-billion-parameter large language model (LLM) developed by Mistral AI based on the transformer architecture. It employs a variant of the standard attention mechanism known as grouped-query attention (GQA), which optimizes performance by computing attention over specific groups of hidden states rather than across all hidden states. This approach enhances both scalability and computational efficiency.

For this study, Tamil Mistral was utilized as one of the primary models. This variant was developed by expanding the original Mistral 7B vocabulary with an additional 50,000 Tamil tokens. It was then pre-trained on a 25GB corpus of Tamil text and subsequently fine-tuned using 400,000 Tamil instruction sets.

**GPT-4o.** GPT-4o is a multilingual and multimodal large language model (LLM) developed by OpenAI. In this study, the GPT-4o API was utilized for augmented text generation. The model is recognized for its capabilities in summarization, translation, and question-answering tasks. Notably, GPT-4o exhibits a deep understanding of the Tamil language, producing clear and coherent Tamil text while effectively following instructions provided in Tamil. Furthermore, it demonstrates strong performance in generating explanatory content for Tamil Sangam literature.

To generate augmented explanations for the poems used in training, both the just-ask and few-shot learning approaches were employed. The newly introduced chat completion API by OpenAI was utilized for this purpose, as it streamlines the text generation process and enables experimentation with inference parameters.

### 3.3 Fine-tuning

Fine-tuning was conducted using QLoRA [25], a Parameter-Efficient Fine-Tuning (PEFT) method that quantizes the pre-trained model to 4-bit precision while employing Low-Rank Adapters (LoRA) for updating model weights. This approach enables the fine-tuning of large-scale models on significantly smaller GPUs while maintaining 16-bit fine-tuning performance despite the 4-bit quantization. The 4-bit NormalFloat (NF4) quantization format was used, with float16 as the compute type.

Both Tamil LLaMA 7B and Tamil Mistral 7B were instruction fine-tuned using two distinct datasets: the curated dataset and the augmented dataset. This process resulted in four separate models, facilitating comparative evaluation.

### 3.4 Hyperparameters

**LoRA.** The parameters below were set up for the LoRA during fine-tuning

- Rank – This parameter controls the intrinsic dimensionality of the low-rank decomposition matrix used. Rank was set to 64 as this value provides moderate complexity which enable capturing diverse pattern without overfitting, this is important as we use small datasets.
- Alpha – This parameter defines the scaling factor for learned weights. Alpha was set to 16 as it is a common choice when rank 64 is used.
- Dropout – This is a regularisation parameter to prevent overfitting by randomly dropping units during training. This was set to 0.1 (10%), as this is a recommended rate for preventing overfitting in a model with moderate number of parameters.

**GPT-4o.** The parameters below were used for augmented text generation

- Max Completion Tokens – This parameter defines the upper bound for number of tokens that can be generated for a completion. This value was not set to allow GPT to create a very detailed description without any limitation on the size.
- Temperature – This parameter defines how creative or random the model is allowed to be and the range for this value is between 0 and 2 (inclusive)[5]. We set this value to 1.0 to have balanced focus and creativity when generating augmented text.

**Tamil LLaMA and Tamil Mistral Fine-tuning.** The parameters below were used for fine-tuning Tamil LLaMA 7B and Tamil Mistral models on Tamil Sangam literature dataset

- **Number of training epochs** – This parameter defines number of training epochs to perform. We set this to 10 as we found during early experiments that loss was still reducing after 5 epochs, so it is expected that setting epochs to10 would allow reducing the loss without overfitting.
- **Gradient accumulation steps** – This parameter defines the number of update steps to accumulate the gradients for, before performing backward/update pass. This value

[5] The temperature range for GPT-4o chat completion API is between 0 and 2 as opposed to the standard range between 0 and 1.

was set to 1, essentially disabling accumulation to allow faster weights updates and to reduce the GPU memory consumption required to store intermediate gradients.
- **Optimizer** – PagedAdamW (32 bit) optimizer was used as it is efficient in handling memory constraints by offloading optimiser states to CPU RAM when not in active use thereby reducing GPU memory pressure.
- **Learning rate** – The initial learning rate is set to 2e-4 as it balances stability and the speed of convergence for AdamW optimizer in transformer-based models.
- **Weight decay** – This is set to 0.001 to provide a very small regularisation which prevents overfitting but allows model to learn effectively.
- **BF16** – BF16 floating-point format was used over FP16 as it is best for large-scale-models like 7B and provides wide range although compromises on precision.
- **Max Gradient Normalization** – This parameter is used for gradient clipping, which limits the size of gradients during backpropagation. This is set to 0.3 to prevent large gradient updates.
- **Warmup ratio** – This parameter allows gradually increasing the learning rate during some initial training steps. This is set to 0.03 to reduce the risk of large initial gradient updates leading the unstable model early on.
- **Group by length** – This parameter indicates whether to group the sequences into batches with same length. This was enabled to save memory and speedup training.
- **LR Scheduler Type** – This parameter defines how learning rate is adjusted. The cosine annealing scheduler was used to as it gradually decreases learning rate following cosine curve and helps escape the local minima.
- **Max Sequence Length** – This is set to None to allow sequences of variable length without truncating or padding them. This combined with Group by length parameter allows optimising the memory without truncation.

**Tamil LLaMA and Tamil Mistral Inference.** The parameters below were used at inference time for Tamil LLaMA and Tamil Mistral fine-tuned models

- **Max Length** – This parameter defines the generated token size. We experimented with 512 and 1024 and settled on 512 as it balances the accuracy and inference speed.
- **Max Return Sequence** – This parameter defines number response to return. This was set to 1.
- **No Repeat NGram Size** – This parameter defines how to handle repeated phrases. This is set to 2 to avoid any repetitive tokens.

### 3.5 Performance Metrics

**BLEU (Bilingual Evaluation Understudy).** This is a method for evaluating similarity between texts. It compares consecutive phrases of generated text with the consecutive phrases of reference text and counts number of matches in a weighted fashion.

$$\text{BLEU} = \text{BP} \times exp\left(\sum_{n-1}^{N} w_n \, log \, p_n\right)$$

Where:

$$\text{Brevity Penalty (BP)}=e1-r/c \quad \text{if } c \leq r$$

**METEOR (Metric for Evaluation of Translation with Explicit Ordering).** This measures the quality of generated text and reference text by considering both accuracy of individual words (precision and recall) and their order within the sentence, the higher the score the better the similarity.

$$\text{METEOR} = F_{\text{mean}} \times (1 - \text{Penalty})$$

Where:

$$F_{\text{mean}} = \frac{(10 \times P \times R)}{(R + 9P)} \text{ and Penalty} = 0.5 \times \left(\frac{\text{chunks}}{\text{matches}}\right)^3$$

**chrF Score (Character-level F-Score).** This evaluates reference and generated text similarity by comparing character-level n-gram overlap, making it suitable for Tamil's complex morphology. The F-score is derived from precision and recall of character n-gram matches.

$$\text{chrF} = (1 - \beta^2) \cdot P + \beta^2 \cdot R$$

Where:

P – Precision R – Recall β – Weight factor

**BERTScore.** This evaluates text similarity using BERT's semantic understanding, offering a more nuanced comparison than word-matching metrics like BLEU or ROUGE. It measures how well the generated text aligns with the reference by considering contextual meaning rather than just word presence. For reference $x$ and generated $\bar{x}$ the precision, recall and F1 scores are

$$P_{BERT} = \frac{1}{N_C} \sum_{i=1}^{N_C} \max_{j \in [1, N_R]} x_i^\top y_j$$

$$R_{BERT} = \frac{1}{N_R} \sum_{j=1}^{N_R} \max_{i \in [1, N_C]} x_i^\top y_j$$

$$F_{BERT} = 2 \times \frac{P_{BERT} \cdot R_{BERT}}{P_{BERT} + R_{BERT}}$$

Where:

C = Set of token embeddings in the generated sentence
R = Set of token embeddings in the reference sentence
$x$ and $y$ = Token embeddings from BERT
$x^\top y$ = Cosine similarity between token embeddings

**Human Evaluation.** Human evaluation assesses quality, coherence, and simplicity. A native Tamil speaker with at least 12 years of formal education rated the generated text on a 0–9 scale based on similarity to the reference text, grammatical correctness, and ease of understanding. A score of 0 indicates poor similarity, grammatical errors, and lack of clarity, while 9 signifies high semantic similarity, coherence, and readability.

### 3.6 Published Models and Datasets

Fine-tuned models and datasets can be found at https://huggingface.co/kodebot/.

## 4 Experimental Results and Discussion

As part of this study, a series of experimental activities were conducted, and the corresponding results were analyzed. The dataset collection and preprocessing were performed on a MacBook Pro with 32GB RAM and an Intel i9 processor. Fine-tuning was conducted using Google Colab, leveraging NVIDIA's state-of-the-art A100 GPU. Due to multiple iterations involving trial-and-error and parameter tuning, the cumulative training time amounted to approximately 40 hours. Following fine-tuning, the trained models were uploaded to Hugging Face's model repository, while the datasets were made available in the Hugging Face dataset repository for further evaluation and accessibility.

### 4.1 Dataset collection and augmentation

The dataset was constructed by extracting data from Thirukkural Karpom and Tamil Drops using Scrapy, followed by preprocessing to remove non-Tamil tokens, irrelevant characters, and structural elements. The cleaned text was manually reviewed and formatted into instruction sets using the LLaMA 2 prompt template.

To generate an augmented dataset, GPT-4o was employed to create explanatory texts for Thirukkural, Akananuru, and Purananuru. Initial few-shot learning experiments, providing three poem-explanation pairs per literary work, yielded reasonable but inconsistent results. Testing various temperature settings led to 1.0, balancing creativity and focus. A zero-shot approach was also attempted, but inconsistencies arose due to GPT-4o's difficulty in accurately retrieving specific verses.

To improve reliability, prompts were refined to include the full poem before requesting an explanation, significantly enhancing output quality. The best-performing approach was adopted for dataset expansion. Generated texts were cleaned, reviewed for accuracy, and reformatted. 42 poems required regeneration at a higher temperature due to unsatisfactory explanations.

The final dataset consisted of two versions: one with only the collected data and another incorporating both collected and augmented content, ensuring enhanced accessibility and contextual understanding of Tamil literary texts.

### 4.2 Fine-tuning of Tamil LLaMA 2 7B and Tamil Mistral 7B

Fine-tuning adapts a pre-trained model for a specific task by adjusting its parameters to enhance performance. Tamil LLaMA and Tamil Mistral, proficient in modern Tamil, were fine-tuned on Sangam literary texts and explanations to better interpret classical Tamil and its historical context.

Two datasets, collected and augmented, were used, resulting in four fine-tuned models. The data was split 90% for training and 10% for testing to evaluate performance effectively.

The fine-tuning was conducted using PyTorch[6] and the Hugging Face transformers library[7]. The BitsAndBytes library[8] was employed to quantize the base models to 4-bit precision, reducing both computational costs and memory requirements. The models were fine-tuned for 10 epochs with a learning rate of 2e-4 using QLoRA, a Parameter-Efficient Fine-Tuning (PEFT) technique. Additional hyperparameter configurations used in the fine-tuning process are discussed in Section 3.4

### 4.3 Inference with Fine-Tuned Models

The fine-tuned models were evaluated by generating explanatory texts for the test dataset, which comprised 10% of the total dataset that was not used during training. Inference was conducted on all six models, including the two base models, using Google Colab with an NVIDIA A100 GPU. The average inference time ranged between 3 to 5 seconds per poem.

To ensure that the generated text was both focused and comprehensive, the max_length parameter was set to 512. Although initial experiments were conducted with max_length set to 1024, the results were found to be more coherent at a length of 512. Additionally, to mitigate repetitive phrases, different values for no_repeat_ngram_size were tested, with a value of 2 yielding the best results. The generated texts were further processed to remove non-Tamil tokens and other non-printable characters to ensure quality.

### 4.4 Performance Evaluation of Fine-tuned Models

The performance of the fine-tuned models was assessed using BLEU, METEOR, chrF, and BERTScore metrics, as summarized in Table 1. Evaluation Metrics with their scores Additionally, human evaluation was conducted to complement automated metrics and provide qualitative insights into the generated explanations.

---

[6] Pytorch. 2025. Pytorch library. *https://pytorch.org/*. Accessed 9 February 2025.

[7] Hugging Face. 2025. Transformers. *https://huggingface.co/docs/transformers/en/index*. Accessed 9 February 2025.

[8] BitsAndBytes. 2025. BitsAndBytes library. *https://huggingface.co/docs/bitsandbytes/en/index*. Accessed 9 February 2025.

**Table 1.** Evaluation Metrics with their scores

| Evaluation Metric | Tamil LLaMA Base | Tamil LLaMA Collected | Tamil LLaMA Augmented | Tamil Mistral Base | Tamil Mistral Collected | Tamil Mistral Augmented |
|---|---|---|---|---|---|---|
| **BLEU** | 0.844 | 5.129 | 3.491 | 0.390 | 5.432 | 4.880 |
| **METEOR** | 0.051 | 0.112 | 0.105 | 0.060 | 0.110 | 0.113 |
| **chrF** | 25.930 | 35.300 | 34.198 | 28.241 | 35.696 | 35.127 |
| **BERTScore** | 0.672 | 0.662 | 0.656 | 0.610 | 0.662 | 0.669 |
| **Human Feedback** | 1.018 | 1.990 | 2.113 | 1.028 | 1.976 | 2.193 |

**BLEU Score.** The results in Table 1 demonstrate that models fine-tuned on the collected dataset significantly outperform their respective base models in BLEU scores, confirming the effectiveness of fine-tuning in improving text generation quality. However, models fine-tuned on the augmented dataset exhibit a decrease in BLEU scores compared to those fine-tuned on the curated dataset. Specifically, Tamil LLaMA's BLEU score dropped from 5.129 to 3.491, while Tamil Mistral's score decreased from 5.432 to 4.880. This decline is likely due to the increased lexical variation introduced by the augmented dataset, as BLEU penalizes deviations from exact phrasing, even when meaning is preserved.

**METEOR and chrF Scores.** The chrF metric follows a similar trend to BLEU, with the base models achieving the lowest scores and the models fine-tuned on the curated dataset performing best. However, METEOR scores remain relatively stable across models fine-tuned with both curated and augmented datasets. This stability suggests that the core semantic meaning of the generated explanations is preserved despite variations in phrasing, as METEOR accounts for synonymy.

**BERTScore.** BERTScore, which evaluates semantic similarity rather than exact lexical matches, exhibits only minor fluctuations between models fine-tuned on curated and augmented datasets. Tamil LLaMA's score slightly declined from 0.662 to 0.656, while Tamil Mistral's score showed a slight improvement from 0.662 to 0.669. This indicates that the augmented dataset introduced some degree of semantic variation, but the generated explanations remain closely aligned with the reference texts.

**Human Feedback vs. Automated Metrics.** Human evaluation results do not fully align with BLEU and chrF scores. While these metrics decreased for models fine-tuned on augmented data, human evaluators preferred Tamil Mistral’s outputs from augmented data over the collected dataset, suggesting enhanced linguistic diversity and expressiveness. This discrepancy underscores a limitation of automated metrics, which measure textual similarity but may not capture fluency and readability improvements.
Overall, fine-tuning improved all models. While the collected dataset yielded the highest BLEU and chrF scores, the performance drop in augmented models reflects greater

lexical diversity rather than quality degradation. Human evaluation confirms that augmented data enriched explanations, making them more natural and, in some cases, preferable.

## 5 Conclusions, Future Work and Ethical Issues

This study successfully developed novel datasets for Tamil Sangam literature, including Thirukkural, Akananuru, and Purananuru, along with detailed explanatory texts. Additionally, augmented datasets were created to assess the impact of data expansion. By fine-tuning Tamil LLaMA and Tamil Mistral, we demonstrated around 25% improvements in generating meaningful and contextually rich explanations, particularly as confirmed by human evaluation. While automated metrics showed minor variations, the augmented models produced more expressive and diverse explanations, validating the effectiveness of fine-tuning in this domain.

Our work establishes a strong foundation for Tamil LLM fine-tuning, demonstrating that domain-specific datasets significantly enhance model performance. This research highlights the potential of expanding training data and experimenting with alternative architectures, but the improvements already achieved confirm the effectiveness of fine-tuning for Tamil literary interpretation.

Future work could focus on further optimizing model architectures, refining instruction formatting, and exploring advanced evaluation metrics to better capture the richness of Tamil literature. Additionally, as Tamil Sangam texts carry deep cultural and historical significance, it is essential to ensure that AI-generated explanations remain accurate, contextually appropriate, and aligned with scholarly interpretations.

This research marks a significant step forward in leveraging AI to interpret and preserve Tamil Sangam literature. By fine-tuning Tamil-specific LLMs, we enhance their ability to generate high-quality explanations, contributing to the broader effort of promoting classical Tamil knowledge through AI-driven methods while maintaining responsible and ethical AI practices.

### 5.1 References

1. Wikipedia contributors. 2025. Tamil language. *https://en.wikipedia.org/wiki/Tamil_language*. Accessed 8 February 2025.
2. Worlddata.info. 2025. Tamil Speaking Countries. https://www.worlddata.info/languages/tamil.php. Accessed 8 Feb 2025.
3. Wikipedia contributors. 2025. Tamil diaspora. https://en.wikipedia.org/wiki/Tamil_diaspora. Accessed 8 Feb 2025
4. Wikipedia contributors. 2025. Sangam Literature. *https://en.wikipedia.org/wiki/Sangam_literature*. Accessed 8 February 2025.
5. Wikipedia contributors. 2025. Thirukkural. https://en.wikipedia.org/wiki/Kural. Accessed 8 February 2025.
6. Wikipedia contributors. 2025. Old Tamil. https://en.wikipedia.org/wiki/Old_Tamil. Accessed 8 February 2025.

7. Balachandran, A. 2023. Tamil-Llama: A New Tamil Language Model Based on Llama 2. *arXiv preprint arXiv:2311.05845*.
8. Prakash, J. and Vijay, A.A.S., 2023. Cross-lingual Sentiment Analysis of Tamil Language Using a Multi-stage Deep Learning Architecture. *ACM TRANSACTIONS ON ASIAN AND LOW-RESOURCE LANGUAGE INFORMATION PROCESSING*, *22*(12).
9. Papineni, K., Roukos, S., Ward, T. and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation *In*: *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*., pp.311–318.
10. Popović, M. 2015. chrF: character n-gram F-score for automatic MT evaluation *In*: *Proceedings of the tenth workshop on statistical machine translation*., pp.392–395.
11. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q. and Artzi, Y. 2019. Bertscore: Evaluating text generation with bert. *arXiv preprint arXiv:1904.09675*.
12. Banerjee, S. and Lavie, A., 2005, June. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization* (pp. 65-72).
13. Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P. and Bhosale, S. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.
14. Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L. and Chen, W., 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.
15. Kudo, T., 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.
16. Taori, R., Gulrajani, I., Zhang, T., Dubois, Y., Li, X., Guestrin, C., Liang, P. and Hashimoto, T.B., 2023. *Stanford alpaca: An instruction-following llama model*
17. Mukherjee, S., Mitra, A., Jawahar, G., Agarwal, S., Palangi, H. and Awadallah, A. 2023. Orca: Progressive Learning from Complex Explanation Traces of GPT-4.
18. Jiang, A.Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D.S., Casas, D. de las, Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L.R., Lachaux, M.-A., Stock, P., Scao, T. Le, Lavril, T., Wang, T., Lacroix, T. and Sayed, W. El 2023. Mistral 7B.
19. Ahuja, S., Tanmay, K., Chauhan, H.H., Patra, B., Aggarwal, K., Del Corro, L., Mitra, A., Dhamecha, T.I., Awadallah, A., Choudhary, M., Chaudhary, V. and Sitaram, S. 2024. sPhinX: Sample Efficient Multilingual Instruction Fine-Tuning Through N-shot Guided Prompting.
20. Joshi, S., Khan, M.S., Dafe, A., Singh, K., Zope, V. and Jhamtani, T. 2024. Fine Tuning LLMs for Low Resource Languages *In*: Institute of Electrical and Electronics Engineers (IEEE), pp.511–519.
21. Khan, S., Tarun, A., Faraz, A., Kamble, P., Dahiya, V., Pokala, P., Kulkarni, A., Khatri, C., Ravi, A. and Agarwal, S. 2024. Chitranuvad: Adapting Multi-Lingual LLMs for Multimodal Translation *In*: *Proceedings of the Ninth Conference on Machine Translation*., pp.839–851.
22. Yamaguchi, A., Villavicencio, A. and Aletras, N. 2024. How Can We Effectively Expand the Vocabulary of LLMs with 0.01GB of Target Language Text?
23. Zellers, R., Holtzman, A., Bisk, Y., Farhadi, A. and Choi, Y. 2019. Hellaswag: Can a machine really finish your sentence? *arXiv preprint arXiv:1905.07830*.
24. Sakaguchi, K., Bras, R. Le, Bhagavatula, C. and Choi, Y. 2021. Winogrande: An adversarial winograd schema challenge at scale. *Communications of the ACM*. 64(9), pp.99–106.
25. Dettmers, T., Pagnoni, A., Holtzman, A. and Zettlemoyer, L. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*. 36.