Make Tamil literature accessible to common Tamil people using Artificial Intelligence

|  |  |
| --- | --- |
| Module Code | OCOM5204M |
| Module Title | Data Mining and Text Analytics |
| Assessment Title | Assessment 2 |
| Student ID | OD22MV |

# Research Hypothesis & objectives

Tamil is one of the longest surviving classical languages spoken by Tamil people in South Asia and Tamil diaspora across the world.

Tamil Sangam literature (also known as the poetry of the noble ones) is the earliest known literature of South India. The Sangam literature era spanned from c.300 BCE to 300CE. Tamil Sangam literature is broadly classified into Akam (அகம், inner) and Puram (புறம், outer). Agam poetry focuses on feelings/emotions of romantic love, ethics, morality, nature and similar ones. Puram poetry focuses on heroic deeds of war and public life. These literatures also serve as historical documents for many notable events and people, with vivid description.

One of the literatures that is most significant of all is Thirukural (திருக்குறள்). Thirukural contains 1330 short couplets (4 words in the first line and 3 words in the second line) which focuses on moral and ethics, and it is one of most translated ancient literature works.

In short, Tamil literature is a treasure trove of poems that reflects vibrant Tamil culture, inclusivity of both men and women in Tamil society, ethical and moral values upheld by them and anecdotes of historical events and people.

Anyone who understand these literatures can learn many good virtues and try to apply them in their life.

The challenge is, Tamil as we know today has changed significantly over time and it is not possible to understand/enjoy ancient Tamil literature by anyone but Tamil scholars. This is mainly due to two reasons.

1. Most words of ancient Tamil are not used in modern Tamil (changes to vocabulary)
2. Metaphors, similes, and imagery used in the text requires historical contextual knowledge to understand the true meaning.

The aim of this research is to make ancient Tamil literature accessible to anyone who can read and understand modern Tamil using suitable Artificial Intelligence tools and techniques.

Currently, the interest in Tamil literature work is dying, mainly because English dominates the internet and almost all technological advancements. This research would bring latest technological advancements to Tamil too, thereby, reviving interest in Tamil literature by making it accessible to common Tamil people.

The outcome of this research would be a chat based LLM that can take a piece of Tamil literature work as prompt and output the explanation in modern Tamil at inference time. The output would reflect correct understanding of metaphors, similes and imagery.

The quality of the outcome would be verified by combining two different approaches.

1. Automatic evaluation using ROUGE ([Lin, 2004](#Lin)) and BLEU ([Papineni et al., 2002](#Papineni)) metrics.
2. Human verification to ensure quality, coherence, simplicity, and preference.

The success and impact of the work will be assessed by conducting a series of experiments with a group of common Tamil individuals (end users). End users would explore Tamil literature using the newly developed chat LLM and their feedback that spans across factors such as comprehension, enjoyability, speed and the quality of explanation would be used as input for next iteration.

Although Tamil scholars are not the primary users of this research outcome, eventually they would also benefit from this work as this would help them to conduct any research in their field quickly and effectively.

# Background

It is undeniable fact that English dominates LLM space. There are many reasons for that and some of them include the following.

1. English is the lingua franca of education, science, technology, international business, and communication, and more.
2. Vast amount of data available in English makes it relatively easy to gather data/corpus for training and validation.

Pre-trained base LLMs come in many shapes and forms, ranging from large model that can perform variety of tasks to small ones that perform a specific task very effectively.

These models also come with different licenses, broadly classified into one of the following.

1. Closed source, proprietary models (e.g. GPT-4 from OpenAI, Gemini from Google and DeepMind)
2. Open source, free for commercial use models (e.g. LLaMA from Meta)
3. Everything in-between

Most of these models were predominantly trained and fine-tuned in English and they work far better for English prompts than prompts in other languages.

It is a massive undertaking to train a language model from scratch. It involves the following major tasks at a high level.

1. Collecting large corpus of data
2. Improving quality of data by removing bias, incorrect information, etc...
3. Training using large cluster of GPUs or a similar hardware.
4. Fine-tune using various feedback mechanisms, including human feedback.
5. Repeat this process until the desired outcome is achieved.

This is time, resource and cost intensive process and not possible by everyone. So, better alternative is to take an existing pre-trained LLM and fine-tune it for the specific task we are interested in using Parameter Efficient Fine Tuning (PEFT) techniques like LoRA ([Hu et al., 2021](#Hu)), QLoRA ([Dettmers et al., 2024](#Dettmers)) or the likes. This approach reduces cost and time massively by tuning selected layers of the model or by introducing additional low-rank layers without modifying existing ones.

One of the notable Auto-regressive transformer base model ([Touvron et al., 2023](#Touvron)) that is of good size yet open source and free for commercial use is Large Language Model Meta AI (LLaMA) 2 ([Meta’s LLaMA license](https://ai.meta.com/llama/license/)).

We can fine-tune this model by incorporating additional Tamil tokens, training with Tamil instruction dataset and then fine-tune further on ancient Tamil poem explanation.

This research work will be built on top of Tamil LLaMA, an excellent work by [Balachandran, 2023](#Balachandran), that fine-tuned LLaMA by incorporating 16,000 additional Tamil tokens and then fine-tuned for instructions in Tamil using a translated subset of Alpaca ([Taori et al., 2023](#Taori)) dataset and OpenOrca ([Lian et al., 2023](#Lian)) dataset.

Although, Tamil LLaMA ([Balachandran, 2023](#Balachandran)) is good at taking Tamil prompts and returning Tamil completion at inference time, it does not seem to have deeper statistical understanding of the Tamil language to explain ancient Tamil literature at expected level.

We now know from the work by [Balachandran, 2023](#Balachandran) that pre-trained LLMs can successfully be fine-tuned for Tamil language. This gives us a strong foundation and high confidence that it is worth exploring this avenue.

# Importance and contribution to knowledge

Nowadays, most research are driven by the prospect of economic success and funding is severely restricted to any research field that is not directly linked to economic success. Literature is often overlooked when it comes to integrating technological progress, partly because it is commonly assumed that literature has minimal impact on the economy. It may seem true, but literature is indirectly linked to economy. Literature that teaches strong discipline, ethical and moral values can have significant indirect impact on economy, realised through low crime rate, hard working, promoting charity and ethical/green businesses. Japan is one such example where their economic success is propelled by strong work ethics and discipline.

This work can have significant impact on education sector too. Teaching strong ethics and disciplines to students at young age would stick with them for life. It can also help students to learn literature without a dedicate teacher, which is often a luxury in developing or underdeveloped countries.

Learning inclusivity and equality expressed in ancient literature help building strong society with a leading example that promotes equality and inclusivity in the following areas

* Between men and women
* Among different religions
* Among people of different social and economic status
* Among people of different race

It might even be possible to discover new meaning of ancient poems that truly reflects original thoughts of the author.

It also helps teaching importance of historical sites, monuments, and natural environments, thereby reducing instances of vandalism, deforestation, and other form of similar abuses, which is a common occurrence in South Asia nowadays.

Many details of the culture, lifestyle, technology, wealth, business, and agriculture of Tamil civilisation are hidden in Tamil literature and this work could be a stepping stone for historians and archaeologists to discover something new that would help understanding Tamil civilisation even better.

There is a possibility that this work could be expanded to other under-represented languages (most non-European languages) in LLMs and other technology areas, thereby offering same benefits to native speakers of those languages that is enjoyed by English speakers.

This work has the potential to ignite inspiration among individuals to integrate emerging technological advancements into traditionally overlooked fields, thereby expanding their accessibility to diverse audiences. By bridging the gap between technology and under explored areas, such as literature or cultural preservation, innovative solutions can be introduced to address needs that were not met before.

Research work doesn't have to be solely directed towards achieving economically prosperous outcomes. They can serve the purpose of educating individuals and fostering a richer cultural environment, like this one.

This research work can also inspire others to engage in research aimed at societal improvements, rather than solely prioritising economically driven research pursuits.

# Pilot Study

Open source LLMs make research and product development in any field/area that you can imagine possible without incurring huge cost and time, usually involved in pre-training. However, the availability of models of various size and capability makes choosing a pre-trained model, for the problem at hand, a significant task. Choosing a right model could result in significant increase in performance and quality while reducing the time and cost involved in prompt engineering and fine-tuning.

To evaluate the feasibility of this task (explaining Tamil poems in modern Tamil), we need a model that is ready to be experimented with. Because of this and the requirement of a model to work with Tamil tokens, the number of LLMs available to select for this task have been reduced significantly.

Tamil LLaMA was chosen as a model for this pilot study because it satisfies both language and ready-to-use requirements. Tamil LLaMA ([Balachandran, 2023](#Balachandran)) was developed by enhancing LLaMA’s tokeniser with 16,000 additional Tamil tokens and then instruct model was developed by fine-tuning with Tamil instruction set.

Both base and instruct models come in two different sizes, 7b and 13b parameters. This means, we have the following 4 models to experiment with

* Tamil LLaMA base (7b and 13b)
* Tamil LLaMA instruct (7b and 13b)

However, during experimentation, we found that the 13b models won’t work with the GPU (NVIDIA Tesla T4, 16GB) used for this work due to GPU RAM being too small to fit 13b parameters during fine-tuning. So, we only used 7b parameter models for this pilot study.

Also, no changes were made to tokeniser as part of this pilot study due to time constraints but this needs to be explored further in the implementation stage.

This pilot study was also very limited in terms of the choice of model’s architecture. LLaMA uses decoder only architecture ([Touvron et al., 2023](#Touvron)) also known as Autoregressive models. Autoregressive models are good at text generation and a good choice of summarisation ([Touvron et al., 2023](#Touvron)), however, if we consider explainer of ancient Tamil poems in modern Tamil as a kind of translation, then decoder only models may not be a great choice and we could do better with the use of encoder. So, for translation tasks, full transformer models like BART ([Lewis et al., 2019](#Lewis)) or T5 ([Raffel et al., 2020)](#Raffel) that implements ground-breaking innovation presented in “Attention is all you need” paper ([Vaswani et al., 2017](#Vaswani)) could be worth exploring.

Again, due to time constraints, in-context learning and other prompt engineering techniques were not explored for this pilot study but this is something that is worth exploring in the implementation stage.

For the dataset, it was not feasible to collect and include entire ancient Tamil poem corpus for the pilot study so, again due to time constraints, we only used Thirukural corpus for instruction fine-tuning.

Thrikural contains 1330 couplets and each having explanation written in modern Tamil by 3 different well respected Tamil scholars. We have created this dataset from [Thirukural Karpom](https://thirukural-karpom.github.io/) website and chosen one of the explanations for each of the couplets.

We then kept 10% of the couplets as holdout dataset for testing and used the remaining 1197 for training and validation with k-fold cross validation. Though this is a very small corpus for training, it is a good starting place.

We have instruction fine-tuned both base and instruct (7b) models with the same dataset. Fine-tuning was done using one of the PEFT methods - QLoRA ([Dettmers et al., 2024](#Dettmers)) method with rank=64, alpha=16 and dropout=10%. NVIDIA Tesla T4, 16GB GPU was used.

We did not experiment with different QLoRA parameter options due to time and compute constraints. Training was done for 300 epochs but we noticed that the training loss was not going down after 175 epochs in both base and instruct models, this is probably due to small training data size.

We have run the inference pipeline for 10 instructions from holdout dataset (used only 10 as it was taking significantly longer time for inference). Optimising inference time is one of the tasks that need to be performed during implementation stage.

Here is one of the results we got:

|  |  |
| --- | --- |
| **Prompt** | **Output** |
| அகர முதல எழுத்தெல்லாம் ஆதி பகவன் முதற்றே உலகு | **Completion:** அகர முதலே என்று கூறுவர், அது மற்றவற்றிலிருந்து வேறுபட்டொல், பகவன் முதற்றே உலகமாகும். அந் உலகமும் ஆதி.  **Reference:** எழுத்துக்கள் எல்லாம் அகரத்தை அடிப்படையாக கொண்டிருக்கின்றன. அதுபோல உலகம் கடவுளை அடிப்படையாக கொண்டிருக்கிறது. |

The quality of inference text was evaluated using ROUGE ([Lin, 2004](#Lin)) score. The ROUGE evaluation score from test dataset was not great, we got 0 for all variations (rouge1, rouge2, rougeL and rougeLsum) of the score.

However, when verified manually, the Tamil text completion (generated text) from instruct model is somewhat related to the input text, although they are not coherent, grammatically correct, contains some unexpected non-Tamil tokens and many repeated tokens or phrases, it is the evidence that this approach is in the right direction.

Although this is not a great result, but based on the understanding of how LLMs work, this is something that we can improve by obtaining larger corpus of data for fine-tuning. The bigger the corpus, the better the result would be.

This is reasonably good enough result from the pilot phase to take it to the next stage.

Notebooks for this pilot study can be found at

1. [Fine-tune Tamil Llama 7b base for Tamil Poem](https://colab.research.google.com/drive/1PVpKWRqAk29tLkPBSaiPVjqdQXY_fnyY?usp=sharing)
2. [Fine-tune Tamil Llama 7b instruct for Tamil Poem](https://colab.research.google.com/drive/1dR-E9BD4dPTqObGHP3c2E2u-thapkNmH?usp=sharing)

Fine-tuned models can be found at

1. [kodebot/tamil-llama-7b-base-tamil-poem-experiment-1.1](https://huggingface.co/kodebot/tamil-llama-7b-base-tamil-poem-experiment-1.1)
2. [kodebot/tamil-llama-7b-instruct-tamil-poem-experiment-2.1](https://huggingface.co/kodebot/tamil-llama-7b-instruct-tamil-poem-experiment-2.1)

Dataset prepared for this work can be found at

1. [kodebot/Thirukural\_tamil\_with\_meaning](https://huggingface.co/datasets/kodebot/Thirukural_tamil_with_meaning)

LLaMA 2 finetuning pipeline for fine-tuning was developed loosely based on great work from [Labonne, 2023](https://mlabonne.github.io/blog/posts/Fine_Tune_Your_Own_Llama_2_Model_in_a_Colab_Notebook.html).

# Programme and methodology

Implementation of this work involves the following key stages

1. Scope & Study
2. Data Collection
3. Model Selection
4. Adapt and align the model
5. Evaluation
6. Deployment
7. Report

The implementation involves the following users and stakeholders

1. Funding team/authority
2. AI Engineer (self)
3. Tamil Scholars (at least 1) and a group of common Tamil people who can interact with the final product and provide feedback (at least 5).

## Scope & Study

In this phase of the implementation, the scope (what it can and cannot do) of the final product will be clearly defined to make sure there are no gaps in the understanding of final product between funding authority and AI Engineer.

We will also identify Tamil scholar(s) who is/are going to help with this work and a formal agreement will be signed to ensure their availability for this this work. Group of common Tamil people (5 to 10) would also be identified to help with this work during evaluation phase.

Also, the detailed cost requirement for this project will be identified and included in the Detailed Implementation Plan, a deliverable of this phase.

## Data Collection

Data collection is a challenge for any non-English corpus due to scarcity of data on the internet. For Tamil poem, we can find some data on the internet (used SketchEngine to discover sites with Old Tamil text) but majority of the data is in the print format.

The data need to be converted to digital format and cleaned up for quality, especially the explanation text as the explanation is written by both scholars and non-scholars.

Another challenge is corpus size. Although the number of ancient poems available in Tamil is large compared to most other languages, it is still very small compared to the size of corpus needed for training/fine-tuning LLMs. We need to up-sample the data. One approach is to take each verse of the poem and its explanation as an individual training record. In addition to this, we can also up-sample by treating different explanations from different scholars of the same poem as an individual training record.

Deliverable of this phase would be a dataset for tokeniser and a dataset for fine-tuning.

## Training and Evaluation

Model selection is one of the key activities. We need to evaluate both Autoregressive (decoder only) models like LLaMA, Gemma, and full transformer models (encoder and decoder) like BART, T5 because we can, in a way, consider summarisation of ancient Tamil poems in modern Tamil as a translation task.

Identified model’s tokeniser may need to be extended by training a new tokeniser with new set of vocabulary and merging it back with model’s original tokeniser. This way, we are enriching the vocabulary of the model rather than replacing it. This technique for LLaMA models is first published by [Cui et al., 2023](#Cui). This approach is suitable for any model that uses Scentenepiece ([Kudo and Richardson, 2018](#Kudo)) tokeniser which LLaMA 2 (([Touvron et al., 2023](#Touvron)) uses.

Experimentation with in-context learning by prompt engineering will be done to see if a fine-tuning is needed and to what extent. It may be the case that once the vocabulary of the tokeniser is enriched with additional Tamil tokens, one or few-shot prompt engineering would give enough context for the model to come up with correct or mostly correct completion.

Even if prompt engineering didn’t give the desired result, we can identify prompt templates those perform better, and use them for fine-tuning. When correct prompt template is used for fine-tuning, there is a higher chance for improving fine-tuning result significantly.

Fine tuning will be done using Tamil instruction set if needed. The results will be aligned and improved with human feedback using Reinforcement Learning with Human Feedback (RLHF) to ensure they are relevant, simple, and correct. We will also make sure they uphold human values like helpfulness, honesty, and harmlessness (HHH) ([Askell et al., 2021](#Askell)). We may also apply Proximal policy optimisation algorithms ([Schulman et al., 2017](#Schulman)) to make the model more aligned with human feedback.

The model will also be evaluated using standard summarisation and translation metrics like ROUGE and BLEU.

This phase will be repeated until the desired result is achieved for quality and other human preferences.

The deliverable from this phase would be a model and one or more prompt templates that can immediately be used for inference tasks.

## Deployment

Once a model with desired quality and performance is developed, it needs to be prepared for deployment.

This may include activities like

1. Preparing the infrastructure for deployment
2. Quantising the model to reduce its size and inference time
3. Getting the deployment pipeline ready for continuous deployment
4. Developing a simple web-based UI for users to interact with

These activities enable us to deploy the model in a way it is usable for end users with no technical knowledge. We should also include a feature that enable end users to share their feedback. Note, this is like the feedback we used during training stage, but the difference is that this is directly from end users and received after the training stage. However, this can be reviewed and included in the next iteration of the model training.

We also need to include other observability features, so we need continuously monitor the performance of the model and tweak model parameters if that is necessary. Deliverable from this phase would be a ready to use model for the end users.

## Report

At this phase, we prepare the technical report that includes the following

1. Technical approach taken to achieve the result seen in the final product
2. Performance of the model against standard benchmarks
3. Any limitations
4. Optionally, user guide to help end users.

We also get sign off from the Funding authority.

# Workplan Diagram

The following Gantt charts shows the project implementation plan over the proposed 6 months (26 weeks) for this work. It also highlights key deliverables from each phase.



Stages from Model Selection to Evaluation are iterative and we may do this many times within the 10 weeks window until the desired result is achieved.

# References

Askell, A., Bai, Y., Chen, A., Drain, D., Ganguli, D., Henighan, T., Jones, A., Joseph, N., Mann, B. and DasSarma, N. 2021. A general language assistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*.

Balachandran, A. 2023. Tamil-Llama: A New Tamil Language Model Based on Llama 2. *arXiv preprint arXiv:2311.05845*.

Cui, Y., Yang, Z. and Yao, X. 2023. Efficient and effective text encoding for chinese llama and alpaca. *arXiv preprint arXiv:2304.08177*.

Dettmers, T., Pagnoni, A., Holtzman, A. and Zettlemoyer, L. 2024. Qlora: Efficient finetuning of quantized llms. *Advances in Neural Information Processing Systems*. **36**.

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*.

Kudo, T. and Richardson, J. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. *arXiv preprint arXiv:1808.06226*.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V. and Zettlemoyer, L. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Lian, W., Goodson, B., Pentland, E., Cook, A., Vong, C. and Teknium 2023. OpenOrca: An Open Dataset of GPT Augmented FLAN Reasoning Traces. *HuggingFace*. [Online]. [Accessed 23 February 2024]. Available from: https://huggingface.co/datasets/Open-Orca/OpenOrca.

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries *In*: *Text summarization branches out*., pp.74–81.

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.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. and Liu, P.J. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*. **21**(1), pp.5485–5551.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A. and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

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 (2023). *URL https://github. com/tatsu-lab/stanford\_alpaca*.

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*.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*. **30**.

# Appendix

The following tools, websites and platforms were used for developing this report

1. Google Search and Google Scholar
2. SketchEngine
3. Weka
4. ChatGPT 3.5
5. HuggingFace
6. Google Colab
7. Github
8. Mendeley Cite

## Google Search and Google Scholar

Google Search and Google Scholar helped immensely to develop this report. Google Search helped me to find relevant information on the topics of interest very quickly and efficiently. It directed me to many high-quality tutorial and other study material especially during the development of pilot study. Couple of features that I use quite often and were very useful for preparing this report was the “Featured Snippets” and “People also ask”. Featured Snippet helped to get the information quickly without even visiting the source site (I only use this information if I trust the source website) and “People also ask” helped collecting related information that I have not thought about. See Image A for one of the examples queries I used.

|  |  |
| --- | --- |
| **Image A** | **Image B** |

Google Scholar was very useful to find any research paper on topics of interest and its related papers via the citations. It was also useful for me to get references for citation in RefMan format (See Image B) that I can import into Mendeley Cite.

## SketchEngine

SketchEngine was another tool that help me hugely to explore Tamil language corpus. I created a corpus in Old Tamil on Sangam literature and used it to get statistics and other interesting information on Old Tamil vocabulary. Although the tokens include English words as well, it helped me to discover websites that contains Tamil texts. Here is a screenshot of the content of the corpus I created in SketchEngine

A screenshot of a computer

Description automatically generated

## Weka

Initially I was pondering on the idea of creating a separate fine-tuned model for each of type of Tamil poems and use classifier to direct the prompt to right LLM. I used Weka to explore classification algorithms. But later I decided to develop one LLM that works with all poems. Weka helped me to prototype classification quickly without spending too much time on the task.

## ChatGPT 3.5

I used ChatGPT 3.5 to gain deeper understanding of some of the technical questions I had. But I was very cautious and doubled checked the findings with other sources. This is because ChatGPT is known to give incorrect information confidently on many occasions.

Here is some of the questions I asked ChatGPT 3.5

1. **Prompt:** Explain the difference between LoRA and QLoRA fine-tuning techiniques, **Output:** Result was not helpful in this case, the version of the GPT I used doesn’t have the knowledge of these techniques.
2. **Prompt:** Explain Scentencepiece tokenisation with reference. **Output:** It gave me a simple explanation of how this tokenisation method works and pointed me to right technical paper.
3. **Prompt:** Explain HHH in RLHF. **Output:** Not helpful, it doesn’t seem to have knowledge on this topic.

## HuggingFace

I used transformer and other libraries from HuggingFace for pilot study. It was very helpful to fine-tune different models, mostly in a model agnostic way. I have also used pre-trained Tamil LLaMA models available in HuggingFace. I first experimented with these model directly with inference feature available in HuggingFace before starting fine-tuning work in Google Colab. I downloaded the model from HuggingFace for fine-tuning and uploaded fine-tuned model back into HuggingFace. I also used HuggingFace to store dataset prepared for this pilot work.

Here is my profile page with models and datasets

A screenshot of a computer

Description automatically generated

## Google Colab

I started this pilot work using Google Colab as it offered free T4 GPU with some usage restriction applied. It also helped me to continue the work from anywhere without needing to be at a computer that is powerful enough to perform the pilot study tasks. However, T4 GPUs, even if you get Pay As you Go compute units, has only 16GB GPU RAM. This means it was not possible to experiment with 13b LLaMA models using Google Colab. Regardless, it was very helpful for me as it allowed me to continue my work from anywhere without many frictions.

## Github

Github is another platform I used for this work, especially to find Tamil datasets. For example, I found [Thirukural Karpom](https://thirukural-karpom.github.io/) website via the search feature in Github. This is the website I used for collecting dataset for pilot study. The source code hosted in Github for this website included a JSON file with all the data I need. I simply extracted the data and prepared prompt template with python’s pandas library.

## Mendeley Cite

Mendeley Cite was very useful for collecting and managing references. I created a new collection in Mendeley for this work and kept adding all the relevant papers using its import feature (I downloaded RefMan format from Google Scholar and imported into Mendeley).

I also used Microsoft Word plugin to cite papers quickly and effortlessly with Harvard style created for University of Leeds (Yes, University of Leeds citation style is predefined in Mendeley).

Here is a screenshot of the collection I created for this wok in Mendeley Cite

A screenshot of a computer

Description automatically generated

I also tried inserting Bibliography for the references page using this tool, but the formatting was not to my liking, so I manually formatted them. One feature that this tool lacks is the ability to cross-reference citing to Bibliography. I have done this manually.

Overall, all these tools helped me work efficiently and allowed completing this report on time. I have never used some of these tools before and now I know how to use these, and I will go back to them whenever needed.