EACL Shared Task - Caste/Immigration Hate Speech Detection(LLM-Based Approach)

Ipshita Mahapatra ***M.E.Computer Science (2022-24)***
BITS Pilani, Pilani Campus, Rajasthan, India
h20220121@pilani.bits-pilani.ac.in

Abstract

The main objective of this EACL Shared task is to develop a model that helps efficiently detect whether a comment on social media, written in a regional language(Tamil in this case) is a hate speech comment. For this purpose, our team developed various models to classify the given data as hate speech or not efficiently. One of the Proposed methods was an LLM-based approach to classify text. Large language models are AI models specifically generated and trained to have human-like conversations and can be further trained to apply to a customized or niche application. For this approach, Tamil-LLaMA, a LLaMA model pre-trained in Tamil language, was used and fine-tuned further for our requirement.

1 Introduction

Large Language Models are one of the most recent advances in the Artificial Intelligence and NLP domain. These are Models characterized by their Scale, size(containing billions of parameters), and capacity to understand and generate human-like texts. Their ability to learn intricate patterns and representations of linguistic context makes them a valuable resource in Research and Development.

These models employ the Transformer design, which altered how we approached language tasks, resulting in crucial models such as BERT. Later models, such as GPT-3 and GPT-4, and demonstrated the enormous potential of learning from large amounts of data without explicit instructions. However, these huge models have significant flaws, such as being owned by corporations and having biases. In response, various smaller open-source models, such as LLaMA and Mistral, have challenged the dominance of huge models. However, due to their limited vocabulary, these smaller models still fail to generate excellent text in languages such as Tamil. Despite these obstacles, language progress is being made.

2 Related Work

Open Source LLMs such as LLaMA allow us to collaboratively work and train these models for specific, custom applications. These LLMs are pre-trained and available for users. Users can further train the model with data specific to their requirement(this step is called fine-tuning of the model).

Tamil-LLaMA is one such example. Tamil-LLaMA is an LLM model obtained by pre-training LLaMA models using dataset for Tamil Language, and further fine-tuned to answer prompts and questions in Tamil efficiently. The approach we plan to use for hate speech detection task, will focus on finetuning Tamil-LLaMA further for effectively classifying hate speech in Tamil.[1]

ToxicBuddy is another model fine-tuned over GPT-2 LLM that acts as an auditing tool. The purpose of Toxic Buddy was to generate non-toxic queries that would lead to chatbots providing toxic or offensive answers. This is crucial to ensure Chatbots don't behave in any undesired manner.[3]

Other models and frameworks such as HARE are also being developed to leverage the advantages that LLMs offer and make Hate detection on social media easier. Being a relatively new field of research, there is ample experimentation happening with LLMs.

3 Methodology

Large Language Models follow two main steps for training -

1. Pre-training

Pre-training exposes the LLM to vast amounts of data, which allows it to learn the intricate patterns of language, contextual relations and nuances. The model learns and understands the structure and semantics of the language and its complexities in this phase.

2. Fine-tuning

The next step is Fine-tuning this pre-trained Model, as per our requirement. This can be done via the following methods -

- Using Task specific Data
- By tuning the Hyperparameters
- Layer Freezing(freeze certain layers to retain previous knowledge)
- Gradient clipping(to prevent extreme changes to the initial model, we limit the size of gradient)
- Domain-specific Pre-training
- Transfer Learning
- Ensemble learning(taking predictions of multiple fine-tuned models into consideration to improve performance)

Tamil-LLama uses the Transfer Learning method, and the model is fine-tuned using Low Rank Adapters(LoRA)

Transfer Learning - This method is effective when we have limited task specific data. Instead of training models from scratch, transfer learning talks of leveraging the pre-trained models and fine-tuning them to our specific requirements with the limited dataset we have for that requirement.

Low Rank Adapters(LoRA) is one example of Transfer learning methods.[2]

The issue with this set-up is that LLMs have model parameters in billions, and $|\Delta W| = |W_0|$. This makes Fine-tuning heavy to store and computationally expensive.

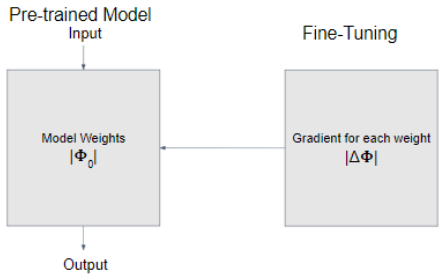

Figure 1: Fine Tuning an LLM

When using LoRA, we work with only a small subset of the model parameters to train the model(while retaining or freezing the initially trained model)

Let our Pre - trained Model = $P_\phi(y|x)$ (Any generic learner, LLaMA2 in case of Tamil-LLaMA)

Initial weights of the pre-trained Model = W_0 Updated weights - W_0 + ΔW Here, we decompose $\Delta\phi$ into B and A, where $W_0 \epsilon R^{dk}$, $B \epsilon R^{dxr}$, $A \epsilon R^{rk}$,rank r = rank of matrix W = min(d,k)

So, now we're dealing with only 2 additional parameters of size much smaller than $|\Delta W$

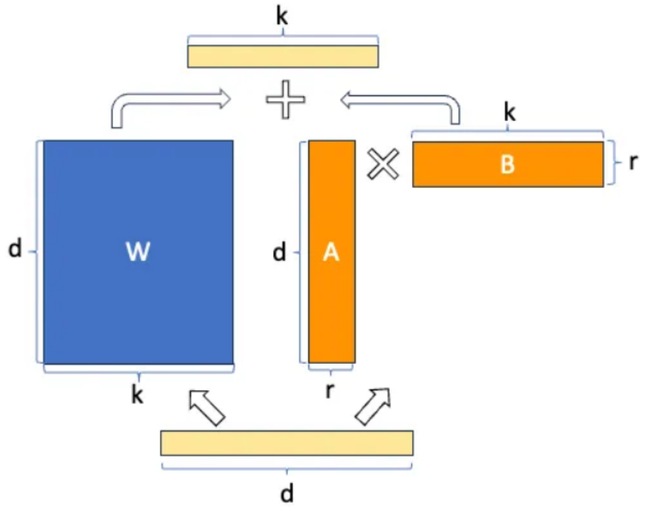

Figure 2: Fine Tuning using LoRA [2]

For the Output h = W_0x, our modified forward pass yields:

h = W_0x + ΔW x = W_0x + BAx

Tamil-LLaMA uses LoRA method to fine tune Tamil-LLaMA-7b(having 7 billion parameters) and Tamil-LLaMA-13b(having 13 billion parameters) to make the model efficient in answering questions in Tamil.

4 Experiments

We further take this base model Tamil-LLaMA-7b, and fine-tune it using LoRA with respect to the data provided for Hate speech detection in Tamil. Since Tamil-LLaMA is pre-trained using the Tamil Language corpus CulturaX, we directly move to the Fine-tuning step.

4.1 Dataset

The dataset provided contains comments from social media(in Tamil and English) and is divided into training, testing and dev data. The format of the dataset is a csv file(index,text,label).

Based on this dataset, we generated data required in the form of

```
<prompt>
### Instruction:
<Instruction string>
### Input:
<Input string>
### Response:
<Response string>
```

This is also called the *Alpaca format* for input to LLMs.

4.2 Training and validation Hyper parameters

Tamil-LLaMA 7B and 13B have been pre-trained using the following parameters -

Configurations	7B	13B
Training Data	12GB	4GB
Epochs	1	1
Batch Size	64	64
Initial Learning Rate	2e-4	2e-4
Max Sequence Length	512	512
LoRA Rank	64	64
LoRA Alpha	128	128
LoRA Target Modules	QKVO, MLP	QKVO, MLP
Training Precision	FP16	FP16

Table 1: Pre-training Hyperparameters

LoRA Rank - relates to the number of trainable parameters from the total parameters(7 billion for 7B, 13 billion for 13B model) Generally ranks should be more than 32. Rank values should be higher if the model is training to learn something more than what it already knows

LoRA alpha - This is the scaling parameter for LLaMA

Tamil-LLaMA was further fine-tuned using the following parameters -

Configurations	7B	13B
Training Data	145k	145k
Epochs	2	1
Batch Size	64	64
Dropout Rate	0.1	0.1
Initial Learning Rate	2e-4	2e-4
Max Sequence Length	512	512
LoRA Rank	64	64
LoRA Alpha	128	128
LoRA Target Modules	QKVO, MLP	QKVO, MLP
Training Precision	FP16	FP16

Table 2: Fine-tuning Hyperparameters

We further used this Tamil-LLaMA pre-trained model, and fine tune it with respect to the following training Parameters -

```
peft_params = LoraConfig(
    lora_alpha=128,
    lora_dropout=0.1,
    r=64,
    bias="none",
    task_type="CAUSAL_LM",
)
```

Here, the Task Type refers to the type of pre-training that one would like to perform. We have selected Causal Learning Model, as is implemented by Tamil-LLaMA too.

4.3 Overall Performance

The performance of the two versions of Tamil-LLaMA LLM model were compared against Random and LLaMA-2 7B models over the IndicSentiment Benchmark for Tamil. From the results, we can see that Tamil-LLaMA 7B and Tamil-LLaMA 13B both outperform the other models.

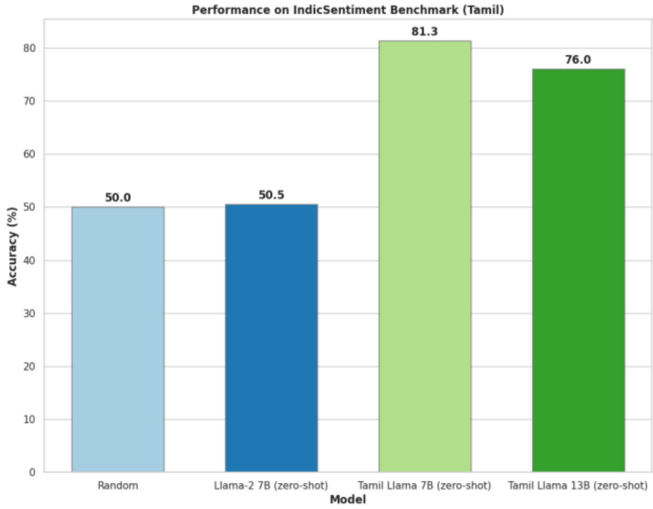

Figure 3: Performance comparison on the IndicSentiment-7B dataset [1]

Presented here are the experimentation conducted and challenges encountered while working on this approach for Hate Speech detection. The experimentations were done on the Tamil-LLaMA-7b model as the base model.

- Working on Google Colab - constant code termination due to disk memory shortage. The general version of Google Colab allows somewhere around 80GB of disk storage, however, pretraining and fine-tuning the Tamil-LLaMA LLM requires more memory(around 50GB more).

- The code was run on a Remote GPU server as well where the kernel issues of the server blocked the code from running

Regretfully, training the Tamil-LLaMA LLM for hate speech detection could not be executed due to system requirement issues.

Acknowledgments

I would like to express my gratitude to Dr. Vinti Agarwal for her invaluable guidance and support throughout the process. Her expertise and insights were helpful in understanding this upcoming topic in the field of AI and ML. I would also like to thank my Institution, BITS Pilani-Pilani Campus, for providing access to the necessary resources and facilities.

References

[1] Abhinand Balachandran, *Tamil-Llama: A New Tamil Language Model Based on Llama 2*. 2023, https://arxiv.org/abs/2311.05845.

[2] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li and Shean Wang, Lu Wang, Weizhu Chen, *LoRA: Low-Rank Adaptation of Large Language Models*. 2021, https://arxiv.org/abs/2106.09685.

[3] Wai Man Si, Michael Backes, Jeremy Blackburn, Emiliano De Cristofaro, Gianluca Stringhini, Savvas Zannettou, Yang Zhang, *Why So Toxic? Measuring and Triggering Toxic Behavior in Open-Domain Chatbots*. 2022, https://arxiv.org/abs/2209.03463.