Exploring the Efficacy of Dynamic Quantization in Large Language Models: A Case Study with FLAN-T5-Base on NLP Tasks

Aditya Dev Koprambil Vinay
Student ID:-231008352
School of Electrical Engineering and Computer Science
Queen Mary University of London London, United Kingdom
a.koprambilvinay@se23.qmul.ac.uk
Project Guide:- Seyedalireza Sanaeekohroudi

***Abstract*—Quantization has become an essential technique for model compression, particularly in the deployment of artificial intelligence (AI) models on edge devices. This study seeks to explore the impact of dynamic quantization on the performance of Large Language Models (LLMs) when applied to various Natural Language Processing (NLP) tasks, such as chat summarization, text classification, sentiment analysis, and reasoning, using the FLAN-T5 model. The research aims to assess how dynamic quantization influences both the accuracy and computational efficiency of the model across these diverse tasks, while also examining the trade-offs between efficiency and performance. The experiments are conducted on four benchmark datasets, each corresponding to a different NLP task, including GSM8K, IMDB, and SAMsum**

***Index Terms*—FLAN-T5, deep learning, dynamic quantization, NLP, LLM, transformer**

I. Introduction

The rapid growth of deep learning models, especially Large Language Models (LLMs), has brought about significant advancements in the field of artificial intelligence (AI). However, the deployment of these models often comes with substantial computational and memory requirements, making them challenging to implement on resource-constrained environments such as edge devices. Model compression techniques have emerged as essential tools to address these challenges, enabling the reduction of model size and computational complexity while striving to maintain performance. Among the various methods available, quantization has gained particular attention due to its ability to reduce model size without altering the underlying architecture. Unlike node pruning, which removes neurons, or matrix decomposition, which approximates weight matrices through lower-rank alternatives, quantization retains the original network structure. This preservation of the model's feature learning capabilities is crucial, especially in scenarios where the model's architecture plays a significant role in achieving high performance on specific tasks.

Quantization works by converting the model's weights from floating-point precision to lower-precision integer values, typically through a scaling factor. This conversion not only reduces the model size but also enhances computational efficiency, particularly in environments with limited hardware capabilities. For instance, by reducing the precision of weights, quantization can significantly decrease the memory bandwidth and storage requirements, thereby enabling faster execution times. This characteristic makes quantization a preferred method for deploying LLMs on edge devices, where both computational resources and energy consumption are constrained. The ability to compress a model while maintaining its architecture makes quantization particularly appealing for applications that require the model to perform complex tasks in real time, such as natural language processing (NLP) on mobile devices.

This research is motivated by the foundational work conducted by previous studies, particularly the study by Ramakrishnan et al. (2021), which performed a cross-platform analysis to evaluate the impact of compressed deep learning models on edge devices. Building on their findings, this study aims to delve deeper into the effects of dynamic quantization on the performance and efficiency of LLMs, with a specific focus on the FLAN-T5 model. By exploring how different NLP tasks—ranging from text classification to machine translation—are affected by quantization, this study seeks to provide a comprehensive understanding of the trade-offs involved. The potential to compress an LLM to just a few megabytes or even kilobytes presents significant opportunities for edge deployment, where reduced inference times and faster execution are paramount. The implications of such advancements extend to various real-world applications, including smart assistants, real-time language translation, and other AI-driven services that rely on processing large volumes of data in constrained environments. This research aims to contribute to the ongoing efforts to create more efficient and scalable AI solutions, paving the way for the widespread adoption of LLMs in diverse, resource-limited settings. In this paper, the aim is to understand how different NLP tasks like:

- Summarising
- Text Classification
- Reasoning
- Sentiment Analyses

In this paper, we will have a literary survey of similar works, we will be discussing the features of the dataset used, the methodologies applied and lastly, the focus will be on discussing the results and future works.

II. Literature Review

Quantization has gained considerable attention as a model compression technique, particularly for deploying large-scale artificial intelligence (AI) models on edge devices with limited computational resources. Traditional quantization methods, such as Quantization-Aware Training (QAT) and Post-Training Quantization (PTQ), have been widely explored for neural networks, especially in the context of computer vision and NLP tasks. QAT integrates quantization into the training process, allowing the model to adapt to lower precision, while PTQ applies quantization after the model has been fully trained. Recent studies highlight the effectiveness of PTQ due to its simplicity and reduced computational cost, making it a preferred choice for LLMs.

In the context of LLMs, quantization has primarily been explored for its ability to maintain model performance while significantly reducing memory and computational requirements. However, most research has focused on the application of quantization to pre-trained models, with limited exploration of its effects on instruction-tuned models. The impact of quantization on tasks such as text summarization, sentiment analysis, and reasoning, particularly when using models like FLAN-T5, remains underexplored. Previous work, such as Jin et al. (2024), has emphasized the need for structured evaluation frameworks to assess the performance of quantized LLMs across diverse benchmarks. This includes the analysis of efficiency, knowledge capacity, and alignment to ensure that quantized models can meet the demands of real-world applications without significant degradation in performance.

In recent years, the increasing complexity of deep learning models has necessitated the development of techniques to optimize these models for deployment on resource-constrained devices. Quantization has emerged as a prominent solution, enabling the reduction of model size and computational requirements by converting floating-point operations into more efficient integer operations. Jacob et al. (2018) propose a comprehensive quantization scheme that allows neural network inference to be conducted using integer-only arithmetic, significantly improving the trade-off between accuracy and latency on mobile devices. This approach is particularly relevant to the current study, which seeks to extend these quantization techniques to (LLMs) like FLAN-T5, with a focus on assessing the impact of dynamic quantization on various NLP tasks. By drawing on the principles established in previous research, this study aims to evaluate how dynamic quantization affects the performance and efficiency of LLMs, offering insights into the potential for deploying these models in edge environments.

The work on DistilBERT by Sanh (2019) offers significant insights into model compression techniques, particularly knowledge distillation, which is highly relevant to the objectives of this study. DistilBERT, a distilled version of BERT, demonstrates that it is possible to significantly reduce the size and computational requirements of large language models while retaining a substantial portion of their performance on downstream tasks. This approach aligns closely with the current research's focus on dynamic quantization, where the primary goal is to compress large language models like FLAN-T5 without compromising their performance across various NLP tasks such as text classification, sentiment analysis, and summarization.

While DistilBERT leverages knowledge distillation to create a smaller and faster model, the current study explores the impact of dynamic quantization on similar NLP tasks. Both studies address the critical need for efficient model deployment on resource-constrained devices, such as mobile phones or edge devices, where computational and memory resources are limited. However, while DistilBERT primarily focuses on maintaining performance through distillation during the pre-training phase, this research investigates how post-training dynamic quantization affects model accuracy and computational efficiency.

The works of Chung et al. (2024) present an in-depth exploration of the effects of instruction-based finetuning on large language models, particularly focusing on models such as Flan-T5 and Flan-PaLM. The study demonstrates how scaling both the size of the model and the number of tasks used in finetuning can significantly enhance the performance of these models across a variety of reasoning tasks. The authors highlight the effectiveness of incorporating instructions during the finetuning process, which allows the model to better generalize across different tasks, thereby achieving state-of-the-art results in benchmarks like Massive Multi-task Language Understanding (MMLU). The versatility of the FLAN-T5 is the main motivation to consider this model for our experiments.

III. Methodologies

A. Dataset

In this sub-section, we will be discussing about the datasets we have used for the experiments. All the datasets used is publicly available and has been used for ethical research purposes only.

1) GSM8K: The GSM8K dataset, introduced by Cobbe et al. (2021), comprises 8,000 meticulously curated grade school math word problems, GSM8K is designed to evaluate

models' proficiency in multi-step reasoning and arithmetic computations. These problems, reflective of the mathematical reasoning expected from students in grades 3–8, present a unique challenge in their integration of complex linguistic structures, contextual comprehension, and arithmetic operations. The dataset's design necessitates a sophisticated interplay between language understanding and mathematical problem-solving, making it an invaluable tool for assessing the cognitive capabilities of LLM models. The inclusion of both problems and their corresponding solutions in the dataset, as shown in Fig.1 facilitates a comprehensive assessment of model performance, encompassing not only accuracy but also the logical coherence of generated solutions.

Problem:
Janet's ducks lay 16 eggs per day. She eats three for breakfast every morning and bakes muffins for her friends every day with four. She sells the remainder at the farmers' market daily for $2 per fresh duck egg. How much in dollars does she make every day at the farmers' market?

Solution:
First, calculate how many eggs Janet sells daily:

$$\text{Eggs sold} = 16 - 3 - 4 = 9 \text{ eggs}$$

Next, determine how much money she makes by selling these eggs:

$$\text{Total earnings} = 9 \times 2 = 18 \text{ dollars}$$

Answer: Janet makes $18 every day at the farmers market.

Fig. 1. Example from the GSM8K Dataset (Adapted from Tarasov & Shridhar (2024)

2) SAMsum: The SAMSum dataset created by Gliwa et al. (2019) is a valuable resource in the field of natural language processing, focusing on the unique challenge of summarizing chat-like conversations. With over 16,000 dialogues, it mimics the informal and fragmented nature of real-life messaging apps. This dataset stands out for its authentic representation of modern communication styles, offering researchers a robust platform to develop and test dialogue summarization models. Its diverse content spans various topics, reflecting the complexity of everyday conversations and pushing the boundaries of what AI can understand and summarize.

Linguists have meticulously crafted and annotated the SAMsum dialogues, ensuring high-quality data for training and evaluation purposes. This attention to detail has made SAMSum a go-to resource for researchers working on conversational AI and summarization technologies. The dataset's unique focus on chat-style communication sets it apart from traditional text summarization tasks, presenting new challenges and opportunities for advancement in natural language understanding. As AI continues to evolve, SAMSum plays a crucial role in improving how machines interpret and condense the nuanced world of human dialogue.

Dialogue:
Wanda: Let's make a party!
Gina: Why?
Wanda: beacuse. I want some fun!
Gina: ok, what do u need?
Wanda: 1st I need too make a list
Gina: noted and then?
Wanda: well, could u take yours father car and go do groceries with me?
Gina: don't know if he'll agree
Wanda: I know, but u can ask :)
Gina: I'll try but theres no promisess
Wanda: I know, u r the best!
Gina: When u wanna go
Wanda: Friday?
Gina: ok, I'll ask

Summary: *Wanda wants to throw a party. She asks Gina to borrow her father's car and go do groceries together. They set the date for Friday.*

Fig. 2. Example from the SAMsum Dataset(Adapted from Yijia-Xiao (2024))

3) IMDB: The IMDB dataset is a key resource for researchers working on sentiment analysis in natural language processing. It contains 50,000 movie reviews from the Internet Movie Database, split evenly between training and testing sets. The review are labeled as positive or negative. This dataset is special because it captures real opinions from moviegoers, showing how people express their thoughts about films. Because of its size and quality, the IMDB dataset has become a benchmark for testing new machine learning algorithms in sentiment analysis. One of the things that makes the IMDB dataset challenging is the length and complexity of its reviews. Movie fans often write detailed opinions with subtle expressions of sentiment, which can be tricky for computers to understand. This has made the dataset incredibly useful for developing and testing various sentiment analysis models. Researchers have used it to explore different ways of processing text, extracting important features, and improving their models. From simple machine learning methods to advanced deep learning techniques, the IMDB dataset has helped push the

Review:
Ned aKelly is such an important story to Australians but this movie is awful. It's an Australian story yet it seems like it was set in America. Also Ned was an Australian yet he has an Irish accent...it is the worst film I have seen in a long time

Lable: *0 neg*

Fig. 3. Example from the IMDB Dataset (Adapted from Group (2024))

field of natural language processing forward, giving us better tools for understanding human opinions in text.

B. LLM Model: FLAN-T5 Base

The FLAN-T5 base model is built upon the T5 (Text-To-Text Transfer Transformer) architecture, a transformer-based model designed for a wide range of natural language processing tasks. The T5 architecture utilizes an encoder-decoder structure, where the encoder processes the input text to generate contextual representations, and the decoder uses these representations to generate the output text. This architecture is highly flexible because it treats every NLP task, such as translation, summarization, and classification, as a text-to-text problem. This means that both the input and output are text sequences, making the model versatile in handling various tasks without requiring task-specific adjustments to the model's architecture.

The encoder in FLAN-T5 is made up of several layers that include self-attention mechanisms and feed-forward neural networks. The self-attention mechanism enables the model to assess the significance of different words in the input text by comparing them to one another, helping it capture relationships and context regardless of how far apart the words are. The decoder similarly utilizes self-attention to generate the output sequence, but it also includes cross-attention layers that focus on the encoded representations from the encoder, ensuring that the generated output is contextually relevant to the input. This architecture enables the model to effectively capture and generate complex language patterns, making FLAN-T5 particularly suitable for tasks that require a deep understanding of context, such as summarization, sentiment analysis, and reasoning.

The FLAN-T5 base model is especially well-suited for our research, which involves evaluating the effects of dynamic quantization across different NLP tasks, including text classification, sentiment analysis, and reasoning. Its ability to handle diverse instructions and perform well on different tasks makes it an ideal candidate for such an exploration. Additionally, its balance between performance and computational efficiency is crucial when applying quantization techniques, which aim to optimize model size and speed without significantly sacrificing accuracy.

C. Model Training and Hyperparameter Tuning

In this section, we will discuss the training strategies we used to train the models on the different datasets. For training the model the PyTorch Paszke et al. (2019) framework were used and the transformer models were imported from the HuggingFace Transformer Library Wolf (2019). The Figure.4. The coding for the experiments were done using the Python3.The Figure.4 gives an overall visual idea of the general flow process for training the model. All the models were trained for varying epochs depending on the dataset they were trained on. The model weights that gave the least validation loss is saved and used for further experimentation's.

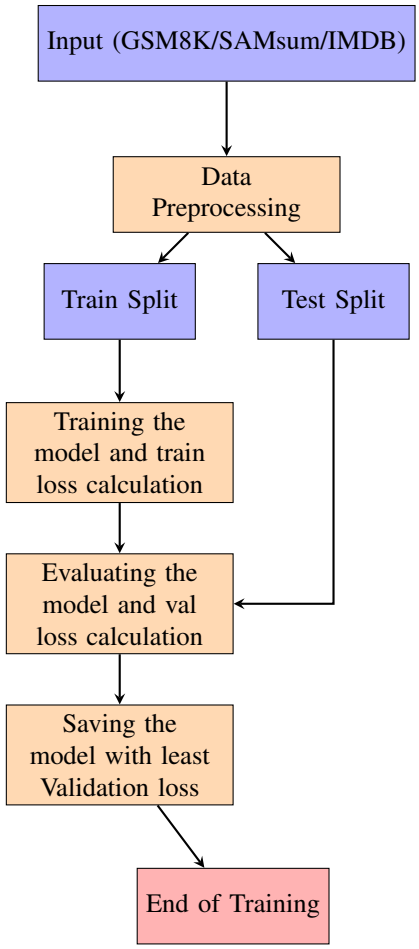

Fig. 4. Flowchart of the training process.

1) FLAN-T5 Base x GSM8K: In this training setup, the model was fed question-answer pair from the dataset as input, and the expected output was the corresponding answer. The questions were tokenized using the T5 tokenizer, and the model was trained to generate the correct answers by minimizing the loss, which was computed as the difference between the model's predicted answers and the actual answers provided in the dataset. The training was conducted over 10 epochs, with a batch size of 8 and a learning rate set at 5×10^{-5} using the AdamW optimizer. A linear learning rate scheduler with warm-up was employed, where the learning rate gradually increased during the first 10% of the total training steps before decreasing linearly.

Gradient clipping was applied to prevent exploding gradients, and early stopping was implemented to avoid overfitting, with the model's performance on a validation set monitored after each epoch. The best model, determined by the lowest evaluation loss, was saved for future use. The graph represented in Figure.5 an insight into the training and validation. From the graph, it is quite evident that the training strategy applied was successful as the training and validation

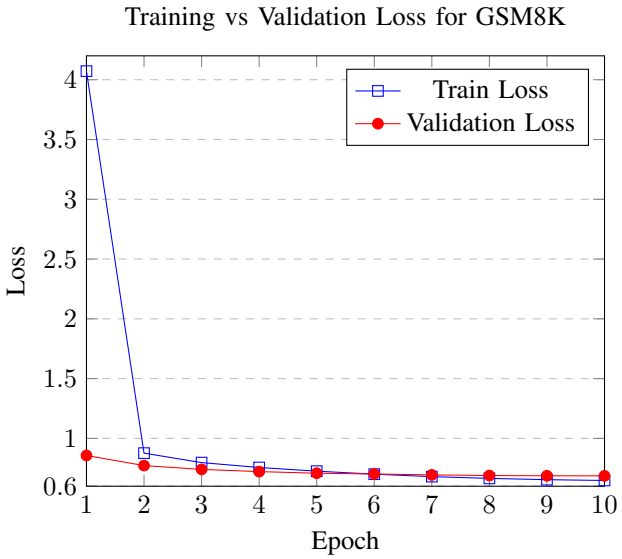

Fig. 5. Train Loss vs Validation Loss over 10 Epochs

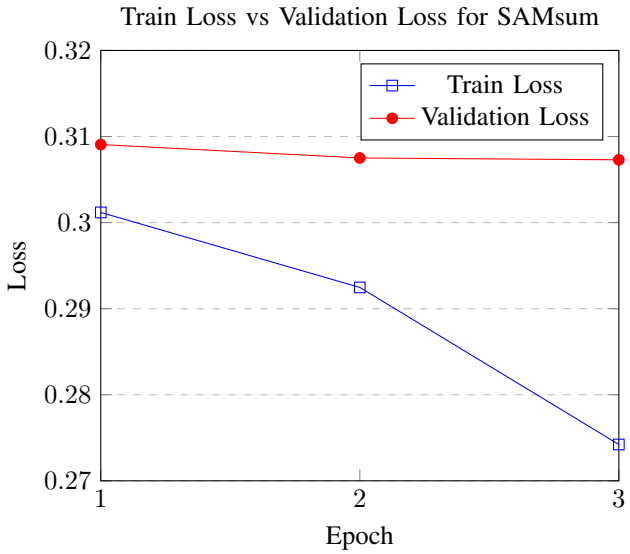

Fig. 6. Comparison of Training Loss and Validation Loss over 3 Epochs

loss is minimised over 10 epochs.

2) FLAN-T5 Base x SAMsum: The FLAN-T5 base model was fine-tuned on the SAMsum dataset, which is specifically designed for dialogue summarization tasks. The dataset consists of thousands of dialogues from various contexts, and each dialogue is accompanied by a human-written summary. In this training setup, the input to the model was the dialogue text, which was tokenized and formatted with the prompt "dialogue:" to guide the model towards generating a summary. The output, or the target label, was the corresponding summary, which was also tokenized and prepared for the model to learn. The training was conducted over 10 epochs with a learning rate set to 1×10^{-4}, using the AdamW optimizer. A linear learning rate scheduler with warm-up was employed, adjusting the learning rate dynamically to optimize the training process. The model was trained on a GPU-enabled environment to leverage hardware acceleration for faster computation.

During the training, both the training and validation losses were monitored to evaluate the model's performance over time. The model's weights were updated using backpropagation based on the computed loss between the predicted summaries and the ground truth summaries. To prevent overfitting, early stopping was employed, saving the best model based on the lowest validation loss observed during training. The model was evaluated on the validation set after each epoch to assess its ability to generalize to unseen data. As you can see in the Figure.6 the model attained convergence in three epochs itself, this is due to the small size of the dataset and the advanced architecture of the model.

3) FLAN-T5 Base x IMDB: The training of the FLAN-T5 base model was conducted using the IMDB dataset, a well-known benchmark for sentiment analysis. This dataset consists of movie reviews labeled as either positive or negative, making it ideal for binary classification tasks. In this training setup, the dataset was preprocessed to filter out examples with text longer than 1000 characters due to compute power restrictions and to ensure that the model could efficiently handle the data. Additionally, the test set was split, with 90% of it added to the training set to enhance the model's learning. The inputs to the model were the movie reviews, framed as "Review: [review text]," while the outputs were the corresponding sentiment labels, either "positive" or "negative." The data was tokenized using the T5 tokenizer, and the model was trained to generate the correct sentiment classification by minimizing the loss between the predicted labels and the true labels provided in the dataset. The input batch size was 32.

The model was fine-tuned over a series of epochs with a learning rate of 1×10^{-5} using the AdamW optimizer, which is known for its stability and effectiveness in training transformer models. A linear learning rate scheduler with a warm-up period was applied, gradually increasing the learning rate during the initial 10% of the training steps before decaying it linearly. The training process spanned several epochs, with the model's performance evaluated on a validation set at the end of each epoch. The best model, determined by the lowest validation loss, was saved for future use. Throughout the training, the model leveraged the FLAN-T5 architecture's ability to handle text-to-text tasks, making it highly effective for sentiment classification tasks like those presented by the IMDB dataset. The Figure.7 gives a visual representation of training of the model.

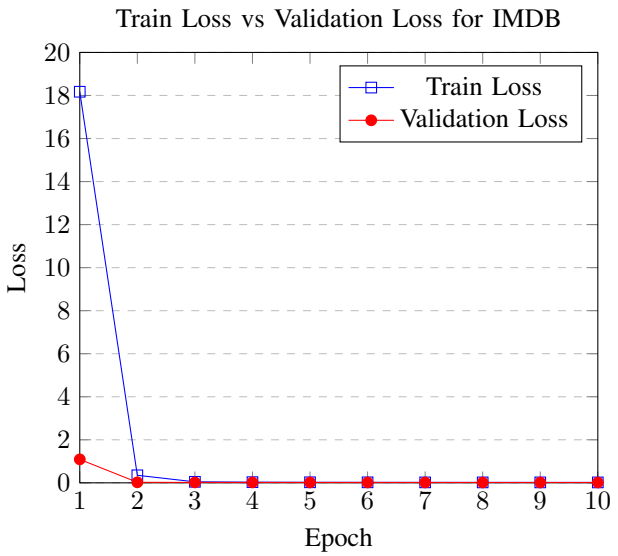

Fig. 7. Comparison of Training Loss and Validation Loss over 10 Epochs

D. Post Training Quantization

Once the three datasets are trained on the FLAN-t5 model the model weights with least validation loss is saved we apply dynamic quantization to a pre-trained transformer model using PyTorch. Initially, we load a pre-trained FLAN-T5 model, which has been fine-tuned on the dataset we are going to infer. The model is evaluated on the validation set of that dataset to determine its original performance and size. The evaluation function calculates the average validation loss by iterating through the dataset, ensuring the model is in evaluation mode to avoid any updates to its parameters. The model size is computed by summing the sizes of all its parameters, and the results are reported in megabytes (MB). This step is crucial for establishing a baseline against which the effects of quantization can be measured.

For the quantization process we converts specific layers of the model, particularly the nn.Linear layers, from full precision (float32) to a lower precision format (int8). This conversion is carried out dynamically, meaning that the weights are converted to int8 only at runtime during inference, while the model's activation layers remain in float32. After quantization, the model is evaluated again on the same validation dataset, but this time on a CPU to ensure compatibility with the int8 format. The code then compares the model size and validation loss before and after quantization. This analysis helps in understanding the trade-off between model efficiency, in terms of memory usage, and any potential degradation in model performance as a result of quantization.

IV. Results and Discussion

A. FLAN-T5-Base x GSM8K

The provided results indicate a significant reduction in the model size after applying dynamic quantization. The original FLAN-T5 Base model, which occupied approximately 143,488.41 MB of memory, was reduced to 6,421.55 MB, resulting in a 95.52% decrease in model size. This substantial reduction demonstrates the efficiency of dynamic quantization in decreasing the memory footprint of large-scale transformer models, making them more suitable for deployment in environments with limited computational resources, such as mobile or edge devices. However, this reduction in size comes at a considerable cost to model performance. The evaluation loss increased from 0.6867 in the original model to 3.3954 after quantization, representing a 394.45% increase. This drastic increase in loss suggests that the quantized model has a

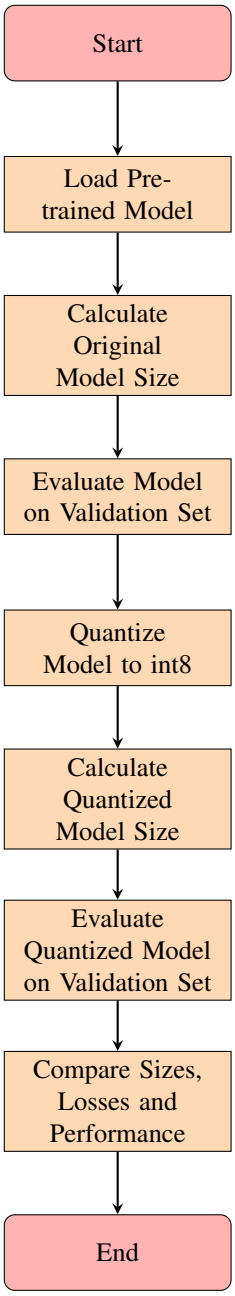

Fig. 8. Flowchart for Quantization process

TABLE I
COMPARISON OF MODEL SIZE AND EVALUATION LOSS BEFORE AND AFTER QUANTIZATION FOR FLAN-T5XGSM8K

Metric	Original Model	Quantized Model
Model Size (MB)	143,488.41	6,421.55
Evaluation Loss	0.6867	3.3954
Size Reduction (%)	95.52%	
Loss Increase (%)	394.45%	

significantly degraded ability to generalize on the validation set, possibly due to the precision loss introduced by the int8 quantization. Therefore, while the memory savings are impressive, the trade-off in model accuracy must be carefully considered, especially in scenarios where model performance is critical. This emphasizes the need for balancing computational efficiency and model accuracy based on the specific application requirements. On further analysis, it was found that although the model was able to understand the basic mathematical question as referred in Figure.1 pre-quantification, once we apply Quantisation it seems that the model was not able to reason well. This observation suggests that quantizing a model meant for reasoning may not be the ideal way to compress it, further experimentation may be required.

B. FLAN-T5 Base x SAMsum

The results from the experiment demonstrate the significant impact of int8 quantization on the T5 model's performance and size. The quantized model achieves a remarkable 95.52% reduction in model size, shrinking from 143,488.41 MB to just 6,421.55 MB. This substantial decrease highlights the efficiency of quantization in compressing large models, making them more suitable for deployment in environments with limited computational resources. However, this compression comes at a considerable cost to model performance. The

TABLE II
COMPARISON OF MODEL SIZE, EVALUATION LOSS, AND ROUGE SCORE BEFORE AND AFTER QUANTIZATION FOR SAMSUM

Metric	Original Model	Quantized Model
Model Size (MB)	143,488.41	6,421.55
Evaluation Loss	0.3076	1.2922
ROUGE-L F1 Score (Mid)	0.4214	0.2142
ROUGE-L Precision (Mid)	0.4664	0.2543
ROUGE-L Recall (Mid)	0.4223	0.2276
Size Reduction (%)	95.52%	
Loss Increase (%)	320.07%	

quantized model exhibits a notable degradation in evaluation metrics. The evaluation loss increased from 0.3076 to 1.2922, a 320.07% increase. Similarly, the ROUGE-L F1 score, a crucial metric for summarization tasks, dropped from 0.4214 to 0.2142. ROUGE (Recall-Oriented Understudy for Gisting Evaluation) Lin (2004) is a set of metrics used to evaluate the quality of text summaries by comparing the overlap between the predicted and reference summaries . The precision and recall metrics also saw significant declines. This suggests that while quantization can drastically reduce model size, it also substantially diminishes the model's ability to generate high-quality summaries. These results underscore the trade-off between model efficiency and performance, highlighting the need for careful consideration when deploying quantized models, especially in tasks where accuracy is critical.

C. FLAN-T5 Base x IMDB

The quantization results reveal a substantial reduction in model size, which highlights the efficiency of int8 quantization. The original FLAN-T5 model size was 990.31 MB, which was reduced by 90.01% to just 98.89 MB post-quantization. This significant size reduction demonstrates the potential of quantization to enable deployment of large language models in resource-constrained environments, such as mobile devices or embedded systems, where memory and storage are limited. However, the compression comes with a severe trade-off in

TABLE III
COMPARISON OF MODEL SIZE, EVALUATION LOSS, AND ACCURACY BEFORE AND AFTER QUANTIZATION FOR IMDB

Metric	Original Model	Quantized Model
Model Size (MB)	990.31	98.89
Evaluation Loss	0.0172	1.8337
Accuracy	0.9452	0.2521
Size Reduction (%)	90.01%	
Loss Increase (%)	10570.07%	
Accuracy Decrease (%)	73.33%	

model performance. The evaluation loss increased dramatically from 0.0172 to 1.8337, a staggering 10,570.07% increase, indicating a substantial degradation in the model's ability to generate accurate predictions. The accuracy also dropped sharply from 94.52% to 25.21%, representing a 73.33% decrease. These results suggest that while quantization greatly reduces the model size, it also significantly compromises the model's effectiveness in sentiment classification tasks. Such a trade-off might be unacceptable in applications where accuracy is critical, emphasizing the need to carefully balance model efficiency and performance when considering quantization.

D. Discussion

From all the experimentation's we can solidly conclude that dynamic quantization can drastically decrease the size of the model, thus enabling them to be deployed on edge devices. A point to be noted here is that when FLAN-T5 base model was trained on GSM8K and SAMsum, post quantization the loss increase by $300\times$ but yet the model did not loose its ability to form meaningful sentences. On closer inspections the model trained with GSM8k was able to reason properly but when it came to the numerical reasoning part the model failed miserably. In case of the SAMsum something similar was observed, although the task at hand was text memorisation the model post dynamic quantization did generate a summarised version of the conversations but it did miss the key elements.

In order to understand the behavior of the llm model post quantization we need to perform further experimentation's. From training the FLAN-T5 model on IMDB dataset on the NLP task of text classification/sentiment analysis task, post quantization observations were not optimistic. The observation strongly suggest that dynamic quantization is not the compression technique for such tasks, but again further wider experiments are required to confirm this.

V. Conclusion

The experiments conducted in this study offer valuable insights into the effects of dynamic quantization on the performance of Large Language Models (LLMs), particularly the FLAN-T5 model, across various NLP tasks. The results show that while dynamic quantization can significantly reduce model size, making it suitable for deployment in resource-constrained environments like edge devices, this often comes at the cost of model performance. For tasks such as reasoning with the GSM8K dataset, text summarization using the SAMSum dataset, and sentiment analysis with the IMDB dataset, quantization led to a notable decline in accuracy and task-specific metrics. The quantized models showed increased evaluation loss and reduced precision, recall, and overall accuracy, highlighting a clear trade-off between computational efficiency and model efficacy. Despite these challenges, dynamic quantization remains a promising approach for compressing LLMs, especially in scenarios where memory and storage limitations are critical. However, the significant performance degradation suggests that further refinement is necessary to strike a balance between efficiency and accuracy. This study underscores the need to consider the specific demands of the application domain when applying quantization techniques, particularly for tasks requiring high accuracy and nuanced understanding, such as reasoning and summarization. Future research should explore alternative quantization strategies or hybrid approaches that could alleviate the performance loss observed with dynamic quantization. Techniques like mixed precision training, which selectively quantizes parts of the model while keeping others in full precision, may offer a more balanced trade-off between model size and accuracy. Additionally, expanding experiments to include other NLP tasks and more complex LLM architectures, such as FLAN-PaLM or GPT-based models, could provide a broader understanding of quantization's impact across different domains.

Moreover, the development of more sophisticated evaluation metrics that go beyond traditional accuracy measures is crucial. These metrics should better assess the model's ability to retain nuanced understanding and reasoning capabilities after quantization. Finally, combining quantization with other compression techniques, such as pruning or knowledge distillation, could result in more efficient and effective model compression, particularly for deployment in resource-constrained environments like edge devices.

References

Chung, H. W., Hou, L., Longpre, S., Zoph, B., Tay, Y., Fedus, W., Li, Y., Wang, X., Dehghani, M., Brahma, S. et al. (2024), 'Scaling instruction-finetuned language models', *Journal of Machine Learning Research* **25**(70), 1–53.

Cobbe, K., Kosaraju, V., Bavarian, M. & et al. (2021), 'Training verifiers to solve math word problems', *arXiv preprint arXiv:2110.14168* .

Gliwa, B., Mochol, I., Biesek, M. & Wawer, A. (2019), Samsum corpus: A human-annotated dialogue dataset for abstractive summarization, *in* 'Proceedings of the 2nd Workshop on New Frontiers in Summarization', pp. 70–79.

Group, S. N. (2024), 'Imdb dataset on hugging face'.
URL: *https://huggingface.co/datasets/stanfordnlp/imdb*

Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A., Adam, H. & Kalenichenko, D. (2018), Quantization and training of neural networks for efficient integer-arithmetic-only inference, *in* 'Proceedings of the IEEE conference on computer vision and pattern recognition', pp. 2704–2713.

Jin, R., Du, J., Huang, W., Liu, W., Luan, J., Wang, B. & Xiong, D. (2024), 'A comprehensive evaluation of quantization strategies for large language models', *arXiv preprint arXiv:2402.16775* .

Lin, C.-Y. (2004), Rouge: A package for automatic evaluation of summaries, *in* 'Text Summarization Branches Out'.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. et al. (2019), 'Pytorch: An imperative style, high-performance deep learning library', *Advances in neural information processing systems* **32**.

Ramakrishnan, R., K V Dev, A., A S, D., Chinchwadkar, R. & Purnaprajna, M. (2021), Demystifying compression techniques in cnns: Cpu, gpu and fpga cross-platform analysis, *in* '2021 34th International Conference on VLSI Design and 2021 20th International Conference on Embedded Systems (VLSID)', pp. 240–245.

Sanh, V. (2019), 'Distilbert, a distilled version of bert: Smaller, faster, cheaper and lighter', *arXiv preprint arXiv:1910.01108* .

Tarasov, D. & Shridhar, K. (2024), 'Distilling llms' decomposition abilities into compact language models', *arXiv preprint arXiv:2402.01812* .

Wolf, T. (2019), 'Huggingface's transformers: State-of-the-art natural language processing', *arXiv preprint arXiv:1910.03771* .

Yijia-Xiao (2024), 'Samsum dataset on hugging face'.
URL: *https://huggingface.co/datasets/Yijia-Xiao/samsum*