Revised problem statement

V Sai Vignesh

Friday 21st February, 2025

1 Problem setting

The following assumptions are made for this problem:

1. There is only one local edge device, and one central cloud server. They communicate through a channel with bandwidth B.
2. We consider only a single request for LLM inference, which originates at the edge device.
3. We will assume that the computation time for a given layer in a given environment as constant. We will represent this by $t_{d,i}^{comp}$ for device and layer i. Similarly, we will use the notation $t_{c,i}^{comp}$ for cloud and layer i.
4. We will only consider quantization just before communication.

2 Adaptive quantization

We want to use token aware adaptive quantization, where tokens with more significant information are given more bits. Such a scheme would look like:

$$\beta(x_i \mid \mathrm{attn}_x, \rho) = \begin{cases} 8, & F(x_i \mid \mathrm{attn}_x, \rho) = 1 \\ 4, & F(x_i \mid \mathrm{attn}_x, \rho) = 0 \end{cases}, \quad \forall i \in [1, N],$$

where x_i denotes the i-th token of x during training and generation processes. $F(x_i \mid \mathrm{attn}_x, \rho)$ discriminates whether the token x_i is an important token based on the most recent attention map attn_x. ρ denotes the important token ratio, i.e., ρ tokens among all tokens are set to important with $F(x_i \mid \mathrm{attn}_x, \rho) = 1$, while the rest are considered unimportant.
We will get the important tokens as follows in this simple setting:

1. Get the most attention scores for each token for a given layer i.
2. Sort these tokens in descending order of attention scores.
3. ρ fraction of tokens with higher attention scores will be given higher bit widths.

The quantization of activations can be represented as:

$$Q(x_i) = \left\lfloor \text{CLIP}\left(\frac{x_i}{\alpha_x}, -2^{\beta(x_i)-1}, 2^{\beta(x_i)-1} - 1\right) \right\rceil, \quad \forall i \in [1, N].$$

CLIP refers to a function that restricts the values of the input to lie within a specified range. This is commonly used in quantization to handle outliers and ensure that all values fall within the representable range of the chosen bit-width. So, the output size of the layer where this sort of quantization is used will be:

$$O_i = [8 * \rho + 4 * (1 - \rho)] * d_{model} * \text{Number of tokens}$$

3 Perplexity differential

Perplexity (PPL) is mathematically defined as the exponentiated average negative log-likelihood of a sequence. This is the mathematical definition of perplexity:

$$\text{PPL} = \exp\left(-\frac{1}{N}\sum_{i=1}^{N} \log P(x_i \mid x_1, x_2, \ldots, x_{i-1})\right)$$

The intuition behind this is as follows: A lower perplexity implies that the model is more confident about its predictions. A higher perplexity means that the model is less certain and assigns lower probabilities to ”correct” tokens.
In the context of LLMs, we will use the softmax probabilities for the right token. Since these probabilities are already conditional, we can directly use these values.
Perplexity is commonly used to evaluate language model quality, and we will also be using this metric to evaluate the effect of various quantizations.
We propose to build a standard, that has the average perplexity difference for different values of ρ. We will use these standards as a constraint in our inference time equation. We can use standard datasets for next token prediction to get the perplexity standards, such as WikiText-103.

4 Statement

$$L_{d,i} = \begin{cases} 1, & \text{if layer i is present on device,} \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

Similarly, we can define:

$$L_{c,i} = \begin{cases} 1, & \text{if layer i is present on the cloud,} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Consider the model to have N layers. If there is a communication happening between the device and the cloud at the end of layer i, it will be represented by

$$C_i = \begin{cases} 1, & \text{if communication between device and cloud happens at end of layer i}, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

We can define t^{comm} as the time taken for communication between the device and the cloud:

$t^{comm} = O_i/B$

Here, O_i is output size at the end of layer i. It will be given by

$$T_{\text{total}} = \sum_{i=1}^{i=N} L_{d,i} * t_{d,i}^{comp} + L_{c,i} * t_{c,i}^{comp} + C_i * t^{comm}$$

subject to

$$\Delta\text{PPL} \leq \Delta\text{PPL}_{\text{max}}$$

Our main goal will be to minimize this inference time under performance constraints.

5 Extensions

These are some of the extensions that can be explored from the original problem statement:

1. Considering the case where there are multiple edge devices in the local edge cluster that can communicate with each other through some means. We would also have to include device constraints in this case.

2. Instead of a single input, we consider a batch of inputs.

3. We consider both training / fine-tuning and inference.

4. We can also model the quantization overhead t^{quant} and the time taken for the request to return output back to the device. This can be represented by a new variable t^{back}. Intra device quantization, to improve inference speed inside the device, can also be considered. This type of quantization can provide a linear speed up in device inference times (the speed of the operations change, but the number of operation do not).

5. Since quantization reduces the output size, it becomes feasible to process inputs in batches more efficiently in a quantized manner compared to non-quantized processing. This can significantly improve throughput by reducing memory bandwidth requirements and leveraging hardware optimizations for low-precision arithmetic.

6 Use cases

These are the advantages of this problem setting:

1. Privacy is maintained. Since the individual tokens themselves are not sent to the cloud server, privacy is better than sending the tokens directly to the server.

2. Maximizing inference time given device, channel and performance constraints.

3. We identify that the major bottleneck in an edge cloud setting is the communication layer between the cloud and the device. By using quantization at this layer, we aim to improve the latency of the operation.