LLM Survey

Piyush Kaul

piyushkaul@ieee.org

March 15, 2025

Overview

LoRA - Low Rank Adaptation of LLM

Objective

1. The trained LLMs are generic. Need to be fined tuned for tasks.
2. Not possible to train LLM without enormous amout of GPUS/Compute power.
3. Weight changes due to fine tuning can be modelled as Low Rank layers in parallel.
4. We only allow the added Low Rank Dcomposition layers to tain.
5. For GPT3, 175 MB, low rank of 2 is adequate. Full rank being quite large.

LoRA - Low Rank Adaptation of LLM

For pre-trained matrix W_0, the augmented networks is defined as.

$$W_0 + \delta W = W_0 + BA \tag{1.1}$$

where $B \in R^{d \times r}$ and $A \in R^{r \times k}$

1. W_0 stays frozen whereas only B and A are trained
2. The basic model weight set W_0 is common across tasks and only B and A need to be switched.
3. Generalization is possible for full fine-tuning as well.
4. Only the attention weights need to be fine tuned. The MLP weight can stay frozen.

LoRA - Results

1. Reduce memory size by 1000 times. 350 GB $\longrightarrow$ 35 MB

LoRA - Layer Decomposition

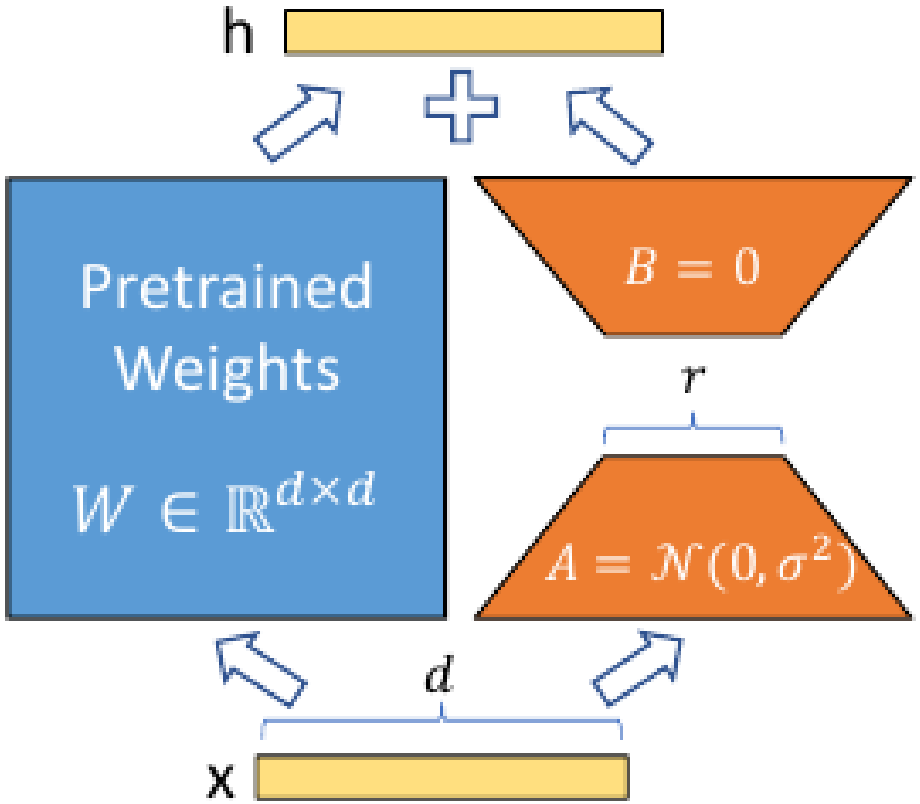

Figure: Enter Caption

QLoRA: Efficient Fine-tuning of Quantized LLM

QLoRA reduces memory requirement from $>$ 700 GB of GPU to $<$48 GB
Following advantages are mentioned

1. 4-bit Normal Float. Better than 4-bit integer and 4-bit float
2. Double Quantization. Quantizing Quant info (saving of approx 3GB for 65 GB)
3. Paged Optimizers. Avoid gradient check pointing spikes.

QLoRA: 4-bit Normal Float

1. Storage as 4b NF. Convert to BF16 for usage.
2. The main limitation of quantile quantization is estimating quantiles.
3. We fix the datatype range [-1,1]. The quantiles of datatype and NN weights need to be matched
4. Estimate 2^{k+1} quantiles for $N(0,1)$ to obtain k-bit quantile regression
5. Normalize its values in [-1, 1] range
6. Quantize input by Re scaling to [-1,1] range

$$q_i = \frac{1}{2}(Qx(\frac{i}{2^k+1}) + Qx(\frac{i+1}{2^k+1}) \tag{2.1}$$

To solve the problem or representing zero exactly, we do this separately for +ve and -ve range.

ZeroQuant-FP

Contribution

1. Negligible degradation from FP16 to FP8
2. FP8 activation along with FP4 weights. LoRC compensates quantization error for w4a8
3. FP4 weights need to be converted to FP8 for inference. Can be done dynamically.
4. Complexity reduction by having scale factors only in power of 2

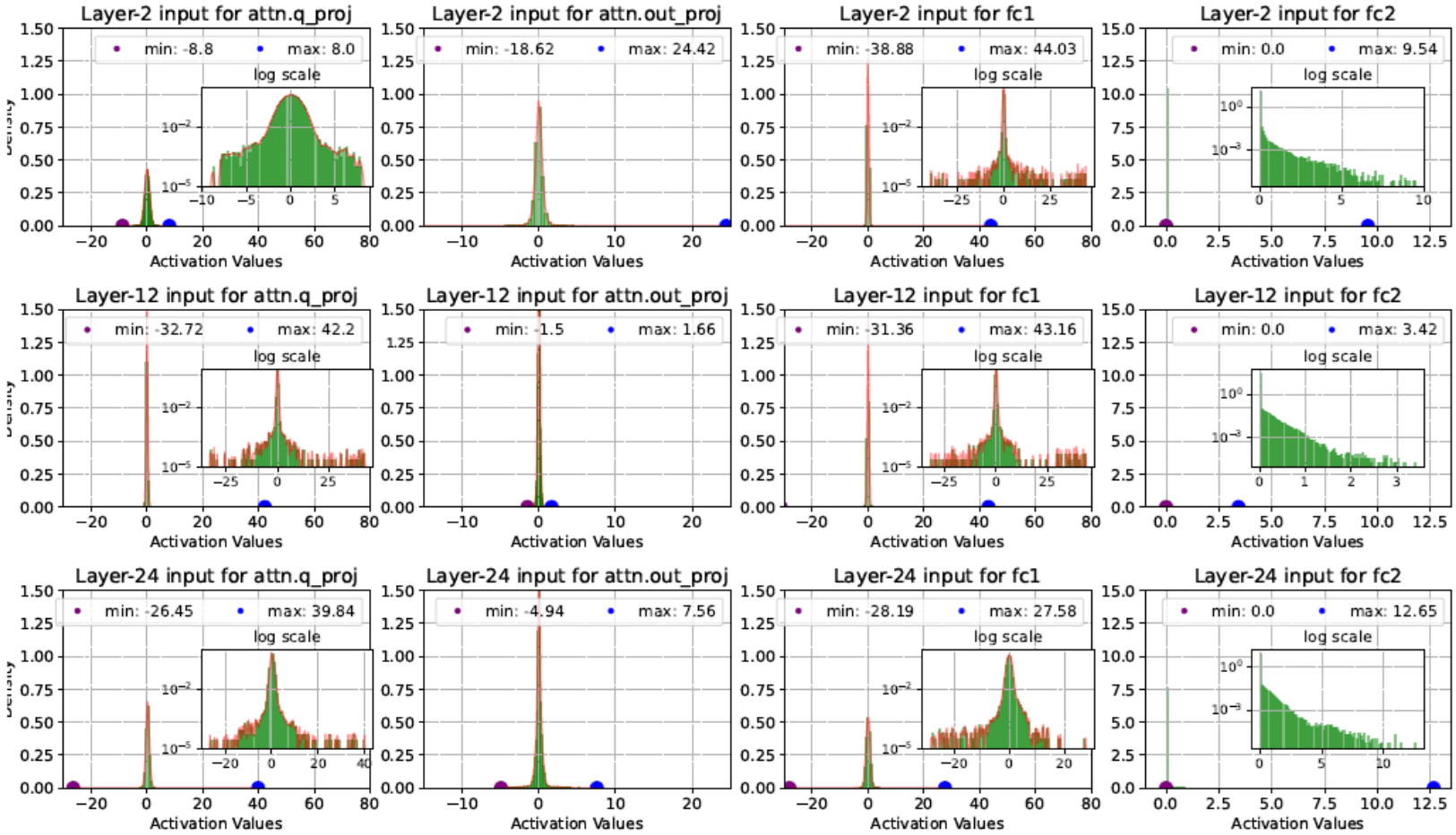

Figure 1: Distribution of Activation values. The top, middle and bottom rows represents the distributions at the 2nd, 12th and final layer of the pretrained OPT-1.3b model. From the left to right columns, they are respectively for the linear modules attn.q_proj (same as attn.k_proj and attn.v_proj), attn.out_proj, fc1, and fc2. The histogram's x-axis ranges from the smallest to largest activation values, while the y-axis denotes their frequency in the dataset. See legend for their minimum and maximum values. Density functions illustrate the probability of different activation values. For more details, please see Section 2.

Figure: Distribution of Activation Values

ZeroQuant-FP - Observations

1. Input to q-proj is Normal
2. The output of q-proj and FC are skewed
3. Integer quantization is not ideal to process skewness

$$Q(x) = INT(X - Z/S) - Z \tag{3.1}$$

4. Integer quantization not ideal for outliers.

INT8 vs FP8

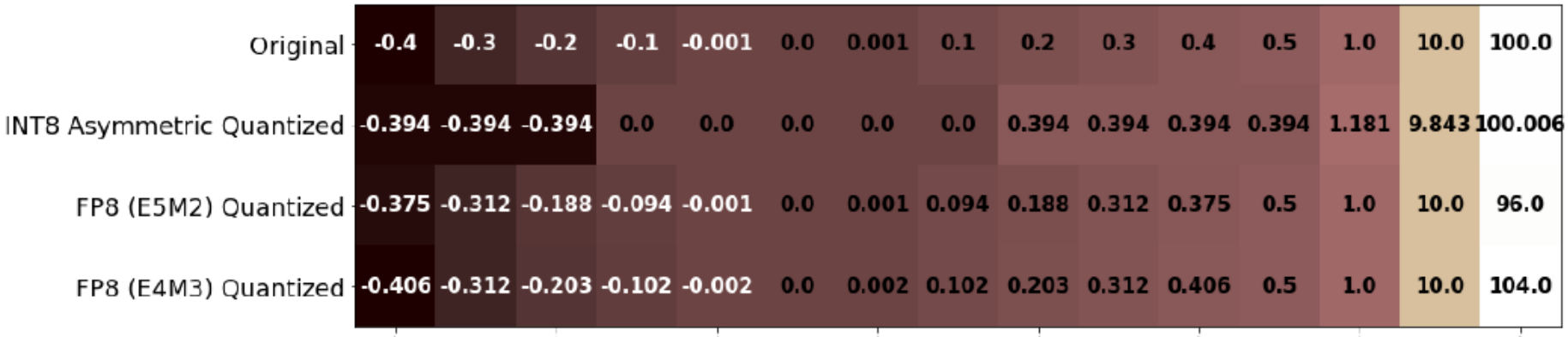

Figure 2: A Contrast between INT8 and FP8 Quantization Methods. The top row displays the original vector in its full-precision form. The subsequent row showcases the vector after quantization through the INT8 Asymmetric approach. The final two rows present values quantized by the FP8 method, utilizing E5M2 and E4M3 formats respectively.

Techniques Used

1. Zero Quant V2 (FGQ and Token Wise Quantization)
2. LoRC
3. Casting FP8 to FP4 with restricted scale

Two methods used for restricting the scale

1. Map to nearest values represented by power of t2 i.e. $\hat{S} = 2^{log_2(S)}$
2. Collect scale in vector $s = [S_1, ..S_x]$. Take maximum and denote by S_{max}. Adjust S_{max}/s

Zero Quant FP

To be represented by power of 2. Then define

$$\hat{S}_i = S_{max}/2^{[log_2(S_{max}/S_i)]} \tag{3.2}$$

Results

Q-type	Weight – Activation	OPT-3b Mean	OPT-3b WIKI/PTB/C4	OPT-7b Mean	OPT-7b WIKI/PTB/C4	OPT-13b Mean	OPT-13b WIKI/PTB/C4	OPT-30b Mean	OPT-30b WIKI/PTB/C4
W16A16	N/A	15.44	14.62/16.97/14.72	11.90	10.86/13.09/11.74	11.22	10.13/12.34/11.20	10.70	9.56/11.84/10.69
W8A8	INT – INT	15.94	14.98/17.49/15.36	12.66	11.20/14.29/12.48	15.94	12.13/19.82/15.86	25.76	14.63/32.90/29.74
	INT – FP	15.85	14.93/17.56/15.05	11.99	10.92/13.24/11.80	11.27	10.16/12.42/11.23	10.69	9.51/11.87/10.71
	FP – FP	15.86	14.97/17.55/15.05	11.99	10.91/13.24/11.81	11.27	10.16/12.42/11.23	10.69	9.51/11.87/10.71
W4A8	INT – INT	16.41	15.39/18.22/15.62	13.18	11.61/15.00/12.92	16.70	12.32/21.21/16.56	24.42	14.80/30.38/28.09
	INT – FP	16.40	15.46/18.23/15.51	12.20	11.13/13.49/11.99	11.34	10.20/12.53/11.30	10.73	9.54/11.91/10.75
	FP – FP	16.29	15.32/18.19/15.35	12.09	10.89/13.44/11.95	11.34	10.16/12.55/11.30	10.72	9.52/11.90/10.75
W4A8 +LoRC	INT – INT	16.38	15.50/18.05/15.59	12.75	11.37/14.33/12.53	15.89	12.06/19.76/15.85	27.20	15.94/34.50/31.16
	INT – FP	16.23	15.40/17.97/15.32	12.13	11.07/13.43/11.90	11.34	10.23/12.49/11.29	10.71	9.48/11.91/10.74
	FP – FP	16.23	15.50/17.92/15.28	12.09	10.96/13.40/11.90	11.33	10.15/12.55/11.29	10.71	9.48/11.90/10.75

Figure: Results

Following Algorithms were tried.

1. RTN
2. GPTQ
3. ZQ-Global
4. ZQ-Local

Following results were obtained

1. GPTQ typically performs better for weight-only quantization, while ZeroQuant (including both ZQ-Global and ZQ-Local) yields superior results for weight and activation quantization.
2. The tested optimization-based methods cannot achieve Class-1 quantization error for either INT4 weight-only or W4A8 quantization with the exception of GPTQ on OPT-30B with weight-only quantization.

Outlier Suppression

1. Channel Wise Shifting and Scaling

$$\tilde{X} = (X - Z) \oslash s \tag{5.1}$$

2. Unified migration pattern

$$\tilde{(}X) \oslash s) W^T + zW^T + b = (\tilde{X} \oslash s) W^T + zW^T + b \tag{5.2}$$

$$= \tilde{X}(w^T \oslash s^T) + (zW^T + b) \tag{5.3}$$

Joint Optimization

Joint Optimization of weights and activations is done by minimizing quantization loss Activation loss is given by

$$min_s[[||Q((X - Z] \oslash s) - (X - Z) \oslash s||_F^2] \tag{5.4}$$

Optimal Brain Surgeon

1. OBS permits a 90%, a 76%, and a 62% reduction in weights over back-propagation with weight decay on three benchmark Monk's problems
2. Of OBS, Optimal Brain Damage, and magnitude-based methods, only OBS deletes the correct weights from a trained XOR network in every case.
3. Optimal Brain Surgeon makes no restrictive assumptions about the form of the network's Hessian, and thereby eliminates the correct weights.
4. OBS does not demand (typically slow) retraining after the pruning of a weight

Optimal Brain Surgeon

Taylor series of error with respect to weights is

$$\delta E = \left(\frac{\partial E}{\partial \mathbf{w}}\right) \cdot \delta \mathbf{w} + \frac{1}{2}\delta \mathbf{w}^T \cdot \mathbf{H} \cdot \delta w + O(||\delta \mathbf{w}||^3) \tag{6.1}$$

With training convergence, 1st term is assumed zero. Third term is also ignored. For pruning we express

$$e_q^T \cdot \delta w + w_q = 0 \tag{6.2}$$

Using the above two We form the Lagrangian

$$\frac{1}{2}\delta w^T \cdot H \cdot \delta w + \lambda(e_q^T \cdot \delta w + w_q) \tag{6.3}$$

Optimal Brain Surgeon

We solve the above Lagrangian to arrive at the optimal weight change

$$\delta w = \frac{w_q}{[\mathbf{H}^{-1}_{qq}]} \mathbf{H}^{-1} e_q \tag{6.4}$$

and corresponding change in loss

$$\delta L_q = \frac{1}{2} \frac{W_q^2}{H^{-1}_{qq}} \tag{6.5}$$

Optimum Brain Surgeon

1. Train a "reasonably large" network to minimum error.
2. Compute H^{-1} .
3. Find the q that gives the smallest saliency $L_q = W_q^2/(2[H^{-1}]_{qq})$. If this candidate error increase is much smaller than E, then the qth weight should be deleted, and we proceed to step 4; otherwise go to step 5. (Other stopping criteria can be used too.)
4. Use the q from step 3 to update all weights (Eq. 5). Go to step 2.
5. No more weights can be deleted without large increase in E. (At this point it may be desirable to retrain the network.

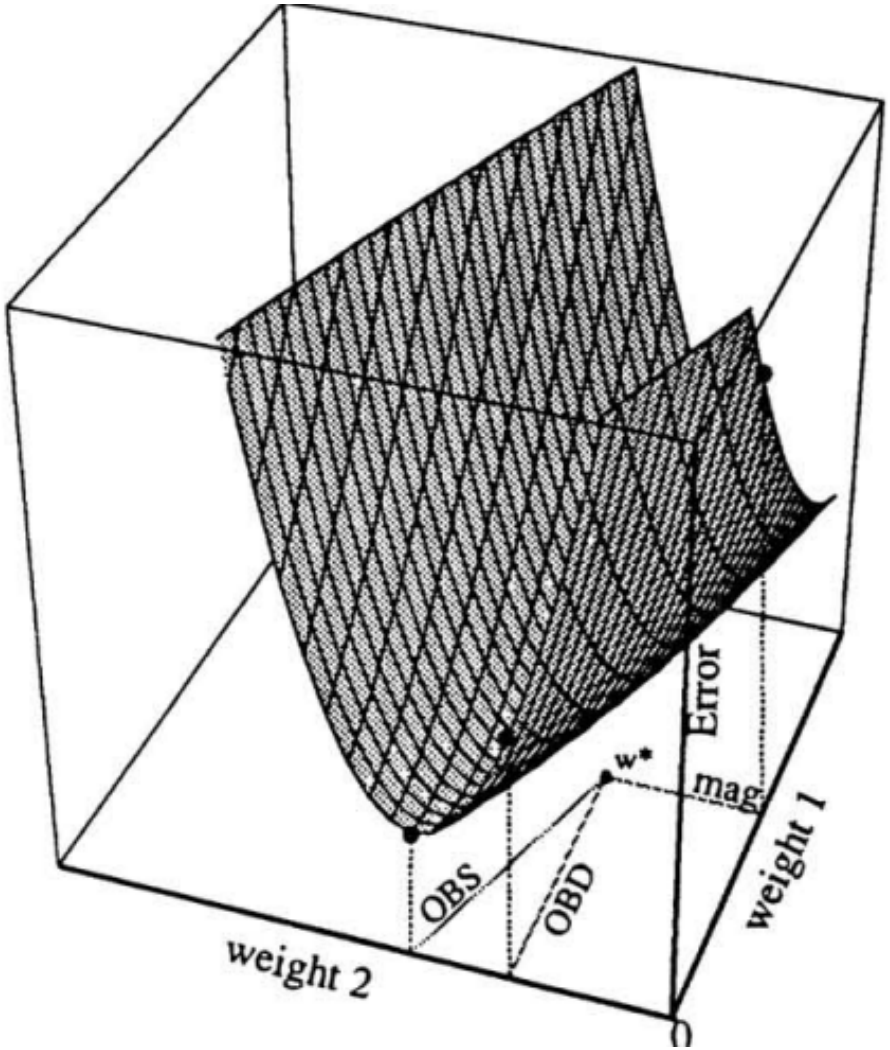

Figure 1: Error as a function of two weights in a network. The (local) minimum occurs at weight w*, found by gradient descent or other learning method. In this illustration, a magnitude based pruning technique (mag) then removes the smallest weight, weight 2; Optimal Brain Damage before retraining (OBD) removes weight 1. In contrast, our Optimal Brain Surgeon method (OBS) not only removes weight 1, but *also* automatically adjusts the value of weight 2 to minimize the error, without retraining. The error surface here is general in that it has different curvatures (second derivatives) along different directions, a minimum at a non-special weight value, and a non-diagonal Hessian (i.e., principal axes are *not* parallel to the weight axes). We have found (to our surprise) that every problem we have investigated has strongly *non*-diagonal Hessians — thereby explaining the improvment of our method over that of Le Cun et al.

Figure: Enter Caption

References I

The End