groupnorm chronicles

latentCall145

April 2024

1 Introduction

Normalization is important because it smoothens out loss landscapes which makes training easier. Image models often use batch normalization but some other models like VQGAN (and VQGAN-based models like Stable Diffusion 1, 2, and SDXL) use group normalization (GN) because GN normalizes each element within the batch separately, resulting in stable training even with small batch sizes.

This writeup describes the GN forward + backward pass with a focus on efficient GPU implementation on the backward pass. I got interested in this project in the first place because mixed-precision convolutions work faster in NHWC format (as opposed to NCHW which is PyTorch's default). The problem is that PyTorch's GN GPU implementation doesn't work for NHWC tensors, so I added NHWC support to GN myself to get the NHWC convolution speedup[1].

2 Forward Pass

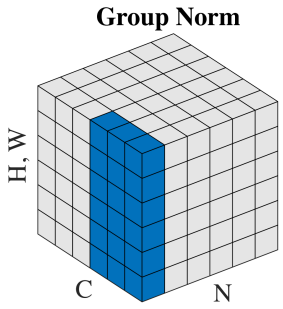

Figure 1: Diagram showing values in image (shaded in blue) that are reduced to one mean/variance during normalization. i drew this cube so many times while coding

[1] Code here

Let the GN forward activation go as the following: $X \in \mathbb{R}^{N \times H \times W \times C}$ is the input. In GN, X is reshaped to $x \in \mathbb{R}^{N \times R \times G \times D}$ (where $R = HW$ and represents the resolution dimension) since that better represents the dimension which is being reduced in the normalization process (along the D and R dimension for x), and thus x will be mentioned instead of X for the remainder of this description. y is the output and is of same shape as x, $\gamma \in \mathbb{R}^C$ is the weights, and $\beta \in \mathbb{R}^C$ is the biases. In a similar way to x, γ, β will be reshaped to $\mathbb{R}^{G \times D}$.

$$\mu_{ng} = \frac{1}{RD} \sum_{r,d} x_{nrgd} \tag{1}$$

$$\sigma_{ng} = \sqrt{\frac{1}{RD} \sum_{r,d} (x_{nrgd} - \mu_{ng})^2} \tag{2}$$

$$\hat{x}_{nrgd} = \frac{x_{nrgd} - \mu_{ng}}{\sigma_{ng}} \tag{3}$$

$$y_{nrgd} = \gamma_{gd} \hat{x}_{nrgd} + \beta_{gd} \tag{4}$$

3 Backward Pass

3.1 Chain Rule Refresher

To preface the backwards derivation, here's a quick reminder of the chain rule: For function $f(x_1(t), x_2(t), ..., x_n(t))$ (note how t is involved in the computation of multiple intermediate functions x_i which then affects f),

$$\frac{\partial f}{\partial t} = \sum_i \frac{\partial f}{\partial x_i} \frac{\partial x_i}{\partial t}$$

3.2 Weight Partials

The backwards derivation for the weight and bias is pretty straightforward. Note that in the below equations, we are looping over n, r (instead of every element of y) since a single element of γ or β affects the output of $n \times r$ elements of y:

$$\frac{\partial f}{\partial \gamma_{gd}} = \sum_{n,r} \frac{\partial f}{\partial y_{nrgd}} \frac{\partial y_{nrgd}}{\partial \gamma_{gd}} \tag{5}$$

$$= \sum_{n,r} \frac{\partial f}{\partial y_{nrgd}} \hat{x}_{nrgd} \tag{6}$$

$$\frac{\partial f}{\partial \beta_{gd}} = \sum_{n,r} \frac{\partial f}{\partial y_{nrgd}} \frac{\partial y_{nrgd}}{\partial \beta_{gd}} \tag{7}$$

$$= \sum_{n,r} \frac{\partial f}{\partial y_{nrgd}} \tag{8}$$

3.3 Activation Partials

The backwards derivation for the input is not as straightforward since nudging any element x_{nrgd} affects every element within its group (because the nudging affects the mean/variance of the group), which in turn affects the output:

$$\frac{\partial f}{\partial x_{nrgd}} = \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \frac{\partial y_{nr'gd'}}{\partial \hat{x}_{nr'gd'}} \frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}} \tag{9}$$

$$= \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} \frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}} \tag{10}$$

Note that the only reason I'm using r', d' for the sums instead of r, d is because r, d specifies the activation input whose partial I want to calculate. We're still summing through the R and D dimension, just under different labels. Focusing on the partials for the normalized variable $\hat{x}$, let's first apply the quotient rule:

$$\frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}} = \frac{1}{\sigma_{ng}^2}(\sigma_{ng} \frac{\partial (x_{nr'gd'} - \mu_{ng})}{\partial x_{nrgd}} - (x_{nr'gd'} - \mu_{ng}) \frac{\partial \sigma_{ng}}{\partial x_{nrgd}}) \tag{11}$$

$$= \frac{1}{\sigma_{ng}}(\frac{\partial x_{nr'gd'}}{\partial x_{nrgd}} - \frac{\partial \mu_{ng}}{\partial x_{nrgd}}) - \frac{1}{\sigma_{ng}^2}((x_{nr'gd'} - \mu_{ng}) \frac{\partial \sigma_{ng}}{\partial x_{nrgd}}) \tag{12}$$

$$= \frac{1}{\sigma_{ng}}(\delta_{r'r}\delta_{d'd} - \frac{\partial \mu_{ng}}{\partial x_{nrgd}}) - \frac{1}{\sigma_{ng}^2}((x_{nr'gd'} - \mu_{ng}) \frac{\partial \sigma_{ng}}{\partial x_{nrgd}}) \tag{13}$$

If you haven't seen the δ_{ij} symbol yet, it's called the Kronecker delta and equals 1 if $i = j$, and 0 otherwise. Thus, $\delta_{r'r}\delta_{d'd}$ equals 1 if $(r', d') = (r, d)$ and 0

otherwise. Now let's unpack the partials for the mean and standard deviation:

$$\frac{\partial \mu_{ng}}{\partial x_{nrgd}} = \frac{1}{RD} \tag{14}$$

$$\frac{\partial \sigma_{ng}}{\partial x_{nrgd}} = \frac{1}{2RD\sigma_{ng}} \sum_{r',d'} 2(x_{nr'gd'} - \mu_{ng}) \frac{\partial (x_{nr'gd'} - \mu_{ng})}{\partial x_{nrgd}} \tag{15}$$

$$= \frac{1}{RD\sigma_{ng}} \sum_{r',d'} (x_{nr'gd'} - \mu_{ng})(\frac{\partial x_{nr'gd'}}{\partial x_{nrgd}} - \frac{\partial \mu_{ng}}{\partial x_{nrgd}}) \tag{16}$$

$$= \frac{1}{RD\sigma_{ng}} \sum_{r',d'} (x_{nr'gd'} - \mu_{ng})(\delta_{r'r}\delta_{d'd} - \frac{1}{RD}) \tag{17}$$

$$= \frac{1}{RD\sigma_{ng}} ((x_{nrgd} - \mu_{ng}) - \frac{1}{RD} \sum_{r',d'} (x_{nr'gd'} - \mu_{ng})) \tag{18}$$

$$= \frac{1}{RD\sigma_{ng}} ((x_{nrgd} - \mu_{ng}) - \frac{1}{RD} \sum_{r',d'} x_{nr'gd'} + \frac{1}{RD} \sum_{r',d'} \mu_{ng})) \tag{19}$$

$$= \frac{1}{RD\sigma_{ng}} ((x_{nrgd} - \mu_{ng}) - \mu_{ng} + \mu_{ng})) \tag{20}$$

$$= \frac{x_{nrgd} - \mu_{ng}}{RD\sigma_{ng}} \tag{21}$$

We're not going to write $\frac{\partial \sigma_{ng}}{\partial x_{nrgd}} = \frac{\hat{x}_{nrgd}}{RD}$ even though it's mathematically equivalent because we don't want to store/load $\hat{x}$ when performing the backward pass on the GPU. Doing so wastes memory and would actually worsen performance as loading/storing values in GPUs (specifically to global memory, which is the GPU equivalent to RAM) is much slower than math operations. Plugging the above partials to $\frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}}$:

$$\frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}} = \frac{1}{\sigma_{ng}} (\delta_{r'r}\delta_{d'd} - \frac{\partial \mu_{ng}}{\partial x_{nrgd}}) - \frac{1}{\sigma_{ng}^2} ((x_{nr'gd'} - \mu_{ng}) \frac{\partial \sigma_{ng}}{\partial x_{nrgd}}) \tag{22}$$

$$= \frac{\delta_{r'r}\delta_{d'd}}{\sigma_{ng}} - \frac{1}{RD\sigma_{ng}} - \frac{1}{RD\sigma_{ng}^3} (x_{nr'gd'} - \mu_{ng})(x_{nrgd} - \mu_{ng}) \tag{23}$$

And now plugging into $\frac{\partial f}{\partial x_{nrgd}}$:

$$\frac{\partial f}{\partial x_{nrgd}} = \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} \frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}} \tag{24}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} (\frac{\delta_{r'r}\delta_{d'd}}{\sigma_{ng}} - \frac{1}{RD\sigma_{ng}} - \frac{1}{RD\sigma_{ng}^3} (x_{nr'gd'} - \mu_{ng})(x_{nrgd} - \mu_{ng})) \tag{25}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} (\frac{\gamma_{gd'}\delta_{r'r}\delta_{d'd}}{\sigma_{ng}} - \frac{\gamma_{gd'}}{RD\sigma_{ng}} - \frac{\gamma_{gd'}}{RD\sigma_{ng}^3}(x_{nr'gd'} - \mu_{ng})(x_{nrgd} - \mu_{ng})) \quad (26)$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{1}{\sigma_{ng}} \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'}\delta_{r'r}\delta_{d'd} + \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} (-\frac{\gamma_{gd'}}{RD\sigma_{ng}} - \frac{\gamma_{gd'}}{RD\sigma_{ng}^3}(x_{nr'gd'} - \mu_{ng})(x_{nrgd} - \mu_{ng})) \quad (27)$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} (-\frac{\gamma_{gd'}}{RD\sigma_{ng}} - \frac{\gamma_{gd'}}{RD\sigma_{ng}^3}(x_{nr'gd'} - \mu_{ng})(x_{nrgd} - \mu_{ng})) \quad (28)$$

3.4 Optimizations

At this point, you can start implementing this on a GPU, but it's going to be relatively slow because each element in Equation 28's sum takes an unnecessary number of operations (RD elements * (3 subtractions and 5 products per element) + $(RD-1)$ additions for the actual sum). To speed this sum up, let's factor out the terms in the loops as much as possible:

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} - \frac{1}{RD\sigma_{ng}} \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} + \frac{\mu_{ng} - x_{nrgd}}{RD\sigma_{ng}^3} \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'}(x_{nr'gd'} - \mu_{ng}) \quad (29)$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} - \frac{1}{RD\sigma_{ng}} \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} + \frac{\mu_{ng} - x_{nrgd}}{RD\sigma_{ng}^3} \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} x_{nr'gd'} + \frac{\mu_{ng}(x_{nrgd} - \mu_{ng})}{RD\sigma_{ng}^3} \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} \quad (30)$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + (\frac{\mu_{ng}(x_{nrgd} - \mu_{ng})}{RD\sigma_{ng}^3} - \frac{1}{RD\sigma_{ng}}) \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} + \frac{\mu_{ng} - x_{nrgd}}{RD\sigma_{ng}^3} \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \gamma_{gd'} x_{nr'gd'} \tag{31}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + (\frac{\mu_{ng}(x_{nrgd} - \mu_{ng})}{RD\sigma_{ng}^3} - \frac{1}{RD\sigma_{ng}}) \sum_{d'} \gamma_{gd'} \sum_{r'} \frac{\partial f}{\partial y_{nr'gd'}} + \frac{\mu_{ng} - x_{nrgd}}{RD\sigma_{ng}^3} \sum_{d'} \gamma_{gd'} \sum_{r'} \frac{\partial f}{\partial y_{nr'gd'}} x_{nr'gd'} \tag{32}$$

Now the loops are really simple. The $\sum_{d'} \gamma_{gd'} \sum_{r'} \frac{\partial f}{\partial y_{nr'gd'}}$ loops have $D(R-1)$ adds for the inner sum, then D mults, then $(D-1)$ adds for the outer sum (so $(RD-1)$ adds and D mults total). The $\sum_{d'} \gamma_{gd'} \sum_{r'} \frac{\partial f}{\partial y_{nr'gd'}} x_{nr'gd'}$ loops have $RD-1$ adds and $RD+D$ mults total.

Even better, the inner loops can be reused for the weight partial calculation. Let $S_{ngd}^{(y)} = \sum_r \frac{\partial f}{\partial y_{nrgd}}$ and $S_{ngd}^{(xy)} = \sum_r \frac{\partial f}{\partial y_{nrgd}} x_{nrgd}$. We can sub these sums in Equation 6:

$$\frac{\partial f}{\partial \gamma_{gd}} = \sum_{n,r} \frac{\partial f}{\partial y_{nrgd}} \hat{x}_{nrgd} \tag{33}$$

$$= \sum_n \sum_r \frac{\partial f}{\partial y_{nrgd}} \frac{(x_{nrgd} - \mu_{ng})}{\sigma_{ng}} \tag{34}$$

$$= \sum_n \frac{1}{\sigma_{ng}} \sum_r \frac{\partial f}{\partial y_{nrgd}} (x_{nrgd} - \mu_{ng}) \tag{35}$$

$$= \sum_n \frac{1}{\sigma_{ng}} (\sum_r \frac{\partial f}{\partial y_{nrgd}} x_{nrgd} - \mu_{ng} \sum_r \frac{\partial f}{\partial y_{nrgd}}) \tag{36}$$

$$\frac{\partial f}{\partial \gamma_{gd}} = \sum_n \frac{S_{ngd}^{(xy)} - \mu_{ng} S_{ngd}^{(y)}}{\sigma_{ng}} \tag{37}$$

Doing the same for Equation 8:

$$\frac{\partial f}{\partial \beta_{gd}} = \sum_{n,r} \frac{\partial f}{\partial y_{nrgd}} \tag{38}$$

$$= \sum_n \sum_r \frac{\partial f}{\partial y_{nrgd}} \tag{39}$$

$$\frac{\partial f}{\partial \beta_{gd}} = \sum_n S_{ngd}^{(y)} \tag{40}$$

And because we haven't done so yet, let's also do Equation 32:

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + (\frac{\mu_{ng}(x_{nrgd} - \mu_{ng})}{RD\sigma_{ng}^3} - \frac{1}{RD\sigma_{ng}}) \sum_{d'} \gamma_{gd'} \sum_{r'} \frac{\partial f}{\partial y_{nr'gd'}} + \frac{\mu_{ng} - x_{nrgd}}{RD\sigma_{ng}^3} \sum_{d'} \gamma_{gd'} \sum_{r'} \frac{\partial f}{\partial y_{nr'gd'}} x_{nr'gd'} \tag{41}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + (\frac{\mu_{ng}(x_{nrgd} - \mu_{ng})}{RD\sigma_{ng}^3} - \frac{1}{RD\sigma_{ng}}) \sum_{d'} \gamma_{gd'} S_{ngd'}^{(y)} + \frac{\mu_{ng} - x_{nrgd}}{RD\sigma_{ng}^3} \sum_{d'} \gamma_{gd'} S_{ngd'}^{(xy)} \tag{42}$$

There are a lot of terms in Equation 42 meaning lots of operations per partial, which is bad because we have to run Equation 42 many times ($NRGD$ times, once for each element of x). By rearranging some terms, we can rewrite Equation 42 in the form:

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + c^{(1)} x_{nrgd} + c^{(2)}$$

This'll greatly reduce the number of operations per element. Here it is:

$$S_{ng}^{(y\gamma)} = \sum_{d'} \gamma_{gd'} S_{ngd'}^{(y)} \tag{43}$$

$$S_{ng}^{(xy\gamma)} = \sum_{d'} \gamma_{gd'} S_{ngd'}^{(xy)} \tag{44}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + (\frac{\mu_{ng}(x_{nrgd} - \mu_{ng})}{RD\sigma_{ng}^3} - \frac{1}{RD\sigma_{ng}}) \sum_{d'} \gamma_{gd'} S_{ng}^{(y)} + \frac{\mu_{ng} - x_{nrgd}}{RD\sigma_{ng}^3} \sum_{d'} \gamma_{gd'} S_{ng}^{(xy)} \tag{45}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + (\frac{\mu_{ng} x_{nrgd}}{RD\sigma_{ng}^3} - \frac{\mu_{ng}^2}{RD\sigma_{ng}^3} - \frac{1}{RD\sigma_{ng}}) S_{ng}^{(y\gamma)} + (\frac{\mu_{ng}}{RD\sigma_{ng}^3} - \frac{x_{nrgd}}{RD\sigma_{ng}^3}) S_{ng}^{(xy\gamma)} \tag{46}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + \frac{\mu_{ng} S_{ng}^{(y\gamma)} - S_{ng}^{(xy\gamma)}}{RD\sigma_{ng}^3} x_{nrgd} - (\frac{\mu_{ng}^2}{RD\sigma_{ng}^3} + \frac{1}{RD\sigma_{ng}}) S_{ng}^{(y\gamma)} + \frac{\mu_{ng}}{RD\sigma_{ng}^3} S_{ng}^{(xy\gamma)} \tag{47}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + \frac{\mu_{ng} S_{ng}^{(y\gamma)} - S_{ng}^{(xy\gamma)}}{RD\sigma_{ng}^3} x_{nrgd} + \frac{-\mu_{ng}^2 S_{ng}^{(y\gamma)} + \mu_{ng} S_{ng}^{(xy\gamma)}}{RD\sigma_{ng}^3} - \frac{S_{ng}^{(y\gamma)}}{RD\sigma_{ng}} \tag{48}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + \frac{\mu_{ng} S_{ng}^{(y\gamma)} - S_{ng}^{(xy\gamma)}}{RD\sigma_{ng}^3} x_{nrgd} - \mu_{ng} \frac{(\mu_{ng} S_{ng}^{(y\gamma)} - S_{ng}^{(xy\gamma)})}{RD\sigma_{ng}^3} - \frac{S_{ng}^{(y\gamma)}}{RD\sigma_{ng}} \tag{49}$$

From here, you can see that $c^{(1)}, c^{(2)}$ have shape $N \times G$ where:

$$c_{ng}^{(1)} = \frac{\mu_{ng} S_{ng}^{(y\gamma)} - S_{ng}^{(xy\gamma)}}{RD\sigma_{ng}^3} \tag{50}$$

$$c_{ng}^{(2)} = -\mu_{ng} c_{ng}^{(1)} - \frac{S_{ng}^{(y\gamma)}}{RD\sigma_{ng}} \tag{51}$$

Putting it all together, here are the partials for the backward pass:

$$S_{ngd}^{(y)} = \sum_r \frac{\partial f}{\partial y_{nrgd}} \tag{52}$$

$$S_{ngd}^{(xy)} = \sum_r \frac{\partial f}{\partial y_{nrgd}} x_{nrgd} \tag{53}$$

$$S_{ng}^{(y\gamma)} = \sum_{d'} \gamma_{gd'} S_{ngd'}^{(y)} \tag{54}$$

$$S_{ng}^{(xy\gamma)} = \sum_{d'} \gamma_{gd'} S_{ngd'}^{(xy)} \tag{55}$$

$$\frac{\partial f}{\partial \gamma_{gd}} = \sum_n \frac{S_{ngd}^{(xy)} - \mu_{ng} S_{ngd}^{(y)}}{\sigma_{ng}} \tag{56}$$

$$\frac{\partial f}{\partial \beta_{gd}} = \sum_n S_{ngd}^{(y)} \tag{57}$$

$$c_{ng}^{(1)} = \frac{\mu_{ng} S_{ng}^{(y\gamma)} - S_{ng}^{(xy\gamma)}}{RD\sigma_{ng}^3} \tag{58}$$

$$c_{ng}^{(2)} = -\mu_{ng} c_{ng}^{(1)} - \frac{S_{ng}^{(y\gamma)}}{RD\sigma_{ng}} \tag{59}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} + c_{ng}^{(1)} x_{nrgd} + c_{ng}^{(2)} \tag{60}$$

4 Fusing Normalization with Activation

One more thing. A lot of models that GN (such as VQGAN-based models) use an activation directly after the normalization. If you implement this with a GN layer followed by an activation layer, values are stored in GPU global memory after normalization, which are then re-loaded to run the activation layer.

As I stated earlier, loading/storing from GPU global memory is really slow, so we can speed up normalization and activation by running both operations in one layer; this technique is called operator fusion[2]. This removes the slow GPU global memory transfer between layers. Operator fusion also saves memory as ML libraries need to store intermediate activations for each layer to calculate gradients, so fusing layers reduces the number of intermediate activations to store.

For the forward pass, we add an activation function $\phi(x)$ to $y_{nrgd} = \gamma_{gd}\hat{x}_{nrgd} + \beta_{gd}$, so y_{nrgd} now becomes:

$$y_{nrgd} = \phi(\gamma_{gd}\hat{x}_{nrgd} + \beta_{gd}) \tag{61}$$

This is trivial to implement on a GPU. For the backward pass:

[2]Operator fusion is one of the most important features (if not THE most important feature) of ML compilers like Triton or XLA.

$$\frac{\partial f}{\partial x_{nrgd}} = \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \frac{\partial y_{nr'gd'}}{\partial \hat{x}_{nr'gd'}} \frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}} \tag{62}$$

$$= \sum_{r',d'} \frac{\partial f}{\partial y_{nr'gd'}} \phi'(\gamma_{gd'}\hat{x}_{nr'gd'} + \beta_{gd'})\gamma_{gd'} \frac{\partial \hat{x}_{nr'gd'}}{\partial x_{nrgd}} \tag{63}$$

Essentially, for every $\frac{\partial f}{\partial y_{nrgd}}$ term you see, you also multiply it with $\phi'(\gamma_{gd}\hat{x}_{nrgd} + \beta_{gd})$, so Equations 52 - 60 become:

$$S^{(y)}_{ngd} = \sum_r \frac{\partial f}{\partial y_{nrgd}} \phi'(\gamma_{gd}\hat{x}_{nrgd} + \beta_{gd}) \tag{64}$$

$$S^{(xy)}_{ngd} = \sum_r \frac{\partial f}{\partial y_{nrgd}} \phi'(\gamma_{gd}\hat{x}_{nrgd} + \beta_{gd}) x_{nrgd} \tag{65}$$

$$S^{(y\gamma)}_{ng} = \sum_{d'} \gamma_{gd'} S^{(y)}_{ngd'} \tag{66}$$

$$S^{(xy\gamma)}_{ng} = \sum_{d'} \gamma_{gd'} S^{(xy)}_{ngd'} \tag{67}$$

$$\frac{\partial f}{\partial \gamma_{gd}} = \sum_n \frac{S^{(xy)}_{ngd} - \mu_{ng} S^{(y)}_{ngd}}{\sigma_{ng}} \tag{68}$$

$$\frac{\partial f}{\partial \beta_{gd}} = \sum_n S^{(y)}_{ngd} \tag{69}$$

$$c^{(1)}_{ng} = \frac{\mu_{ng} S^{(y\gamma)}_{ng} - S^{(xy\gamma)}_{ng}}{RD\sigma^3_{ng}} \tag{70}$$

$$c^{(2)}_{ng} = -\mu_{ng} c^{(1)}_{ng} - \frac{S^{(y\gamma)}_{ng}}{RD\sigma_{ng}} \tag{71}$$

$$\frac{\partial f}{\partial x_{nrgd}} = \frac{\gamma_{gd}}{\sigma_{ng}} \frac{\partial f}{\partial y_{nrgd}} \phi'(\gamma_{gd}\hat{x}_{nrgd} + \beta_{gd}) + c^{(1)}_{ng} x_{nrgd} + c^{(2)}_{ng} \tag{72}$$