GraVAC: Adaptive Compression for Communication-Efficient Distributed DL Training

Sahil Tyagi and Martin Swany

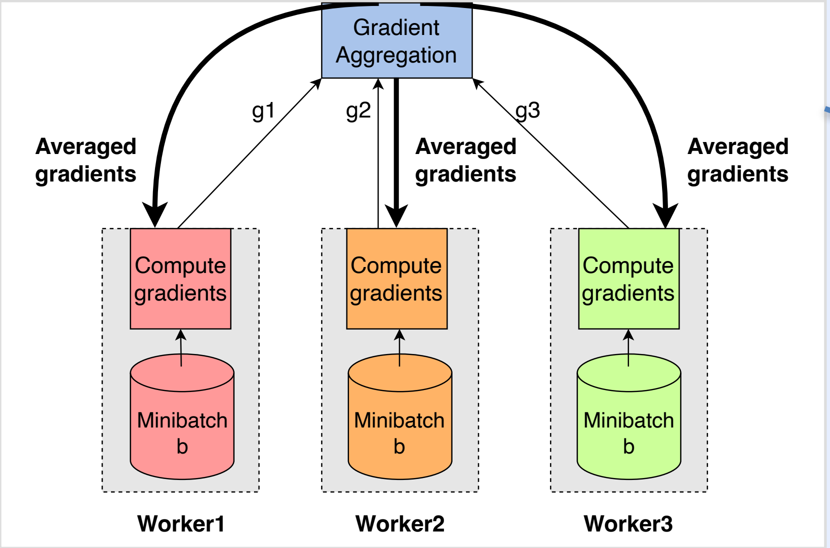
Need for Distributed Training

- Size of deep learning (DL) models has grown exponentially in the last 5 years:
  - **2018**: GPT-1 (100M+), BERT (340M+)
  - **2019**: Transformer-XL (275M+), GPT-2 (1B+)
  - **2020**: BART (140M+), Turing-NLG (17B+)
  - **2021**: ViT (630M+), DALL-E (12B+)
  - **2022/2023**: Stable Diffusion (890M+), GPT-3.5 (1.3B+, 6B+ and 175B+)

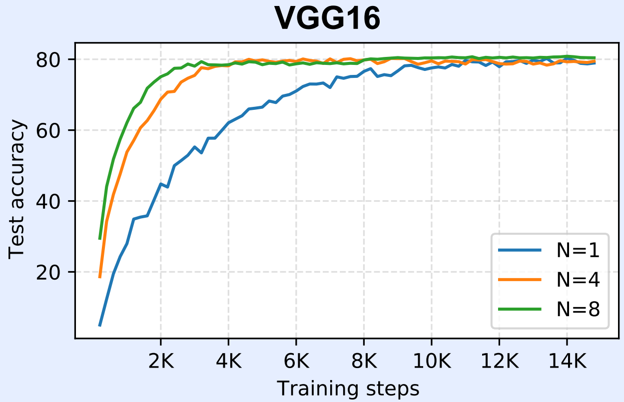
Assuming **single-precision** floats

# parameters	Model-size
10^6	4 MB
10^7	40 MB
10^8	400 MB
10^9	4 GB

Distributed Data-Parallel (DDP) Training



$$w_{i+1} = w_i - \eta \frac{1}{N} \sum_{n=1}^{n=N} \frac{1}{|b|} \sum_{j \in b} \frac{\partial}{\partial w_i} \mathscr{L}(x_{(j,n)}, w_i)$$

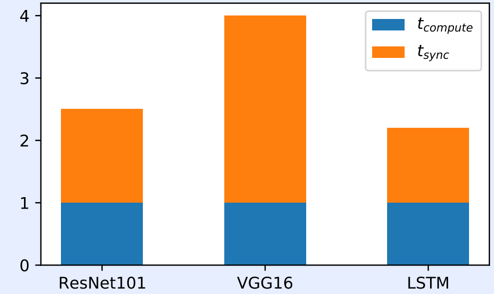


DDP training challenges

- Each training-step time attributed to IO overhead, loss and gradient computation and gradient synchronization

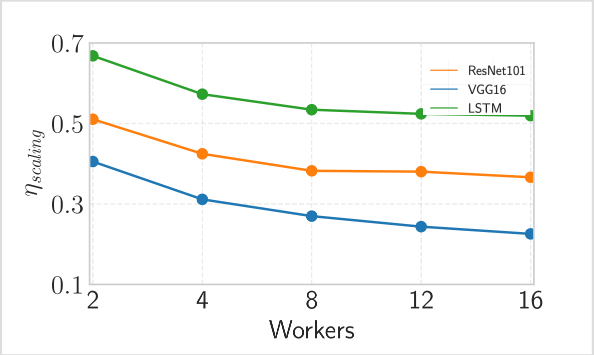
$$t_{step} \approx t_{compute} + t_{sync} + t_{IO}$$

main bottleneck!



- The parallelizability of a job can be measured from its **scaling efficiency**

$$\eta_{scaling} = \frac{T_N}{N \cdot T_1}$$

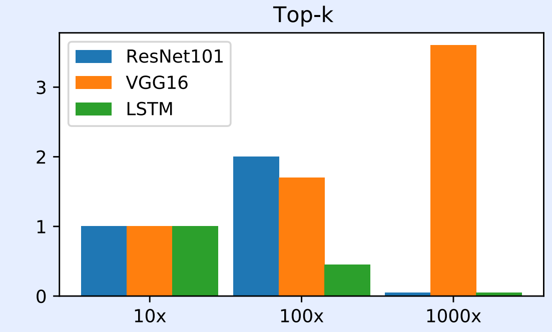


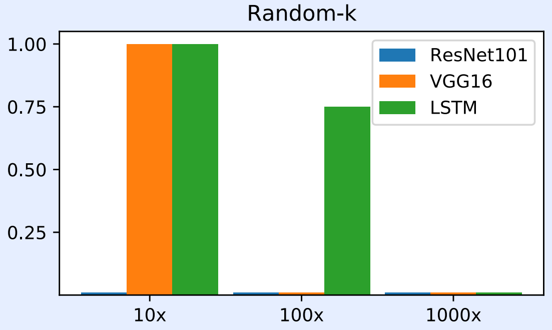
Gradient compression for DDP training

- Gradient compression alleviates communication bottleneck and speeds up training
- **What should be the ideal compression compression factor (CF) with lossy compression?**
  - Reduces tensor volume to communicate
  - Should not trim too much gradients
  - Has acceptable compression overhead

Statistical aspect

Parallel aspect





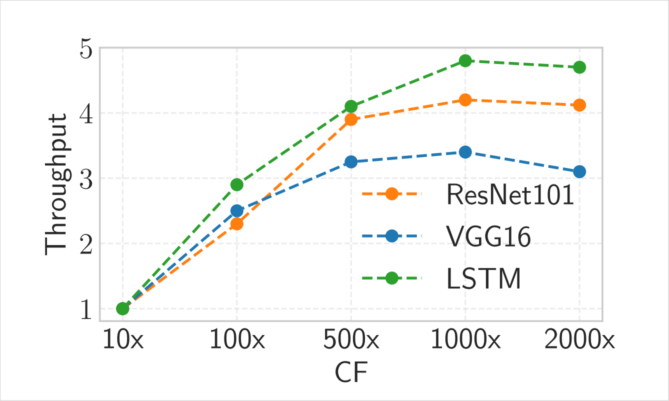
Parallel aspect of Gradient Compression

- To improve scaling efficiency, lossy methods reduce communication but introduce additional compression overhead

$$t_{step}^{(cf)} \approx t_{compute} + t_{IO} + t_{sync}^{(cf)} + t_{compress-decompress}^{(cf)}$$

depends on CF 'cf' and collective op

depends on compression method and 'cf'



Statistical aspect of Gradient Compression

- **Does information loss in compression correlate to model convergence?**
- **Can be seen from prior and post-compression gradients (BC and AC)**

$$g_{ef}^{(i)} = g_{0}^{(i)} + \text{residual}^{(i-1)} \quad \text{and} \quad g_{cf}^{(i)} = C[g_{ef}^{(i)}] \quad \Longrightarrow \quad \text{residual}^{(i)} = g_{ef}^{(i)} - g_{cf}^{(i)}$$

error-feedback critical for convergence

original gradients

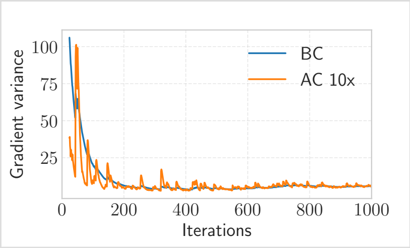
compressed gradients

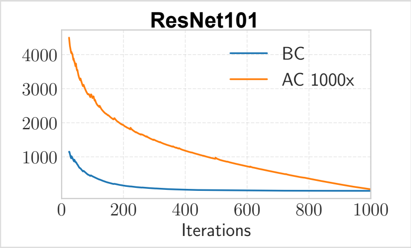
Compressor op

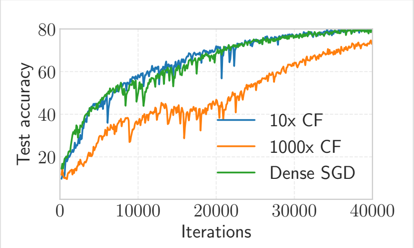
Is there an empirical indicator to measure this information loss?

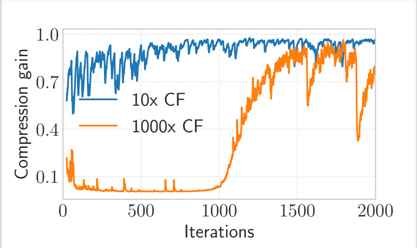
$$\text{Compression gain} = \frac{\mathbb{E}[||g_{cf}^{(i)}||^2]}{\mathbb{E}[||g_{ef}^{(i)}||^2]}$$

higher compression can degrade model convergence or require more training!







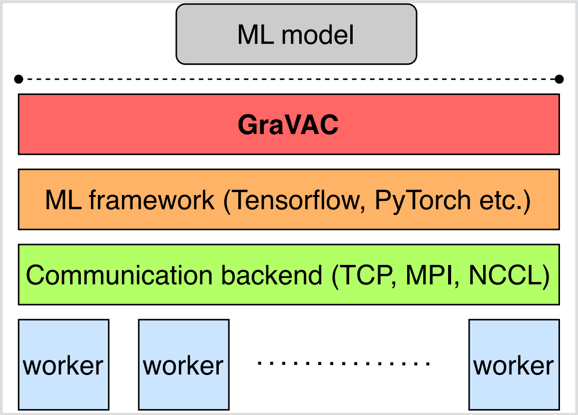


GraVAC's approach

How to choose a CF that considers both the parallel and statistical aspect of gradient compression in DDP training?

$$T_{compression} = T_{system} \times \text{Compression gain}$$

It would be optimal to use low compression in early training phase and higher compression as the model converges and gradients become smaller!



GraVAC = Gradient Variance-based Adaptive Compression

GraVAC's Adaptive Compression

Parameters: CF exploration space $[\mathbf{cf_{min}}, \mathbf{cf_{max}}]$, window-size $\mathbf{w}$, compression step-size $\mathbf{c_s}$, gain threshold ϵ, gain/compression throughput saturation threshold ω

$$g_o^{(i)} = \nabla f(x^{(i)}, w_i)$$
$$g_{min}^{(i)} = C(g_o^{(i)}, cf_{min})$$
$$\delta_{min} = \frac{||g_{min}^{(i)}||^2}{||g_o^{(i)}||^2}$$

$\longrightarrow$

$$g_c^{(i)} = C(g_{min}^{(i)}, c_s)$$
$$\delta_c = \frac{||g_c^{(i)}||^2}{||g_o^{(i)}||^2}$$

effectively compresses original gradients to CF $\mathbf{cf_{min}} \cdot \mathbf{c_s}$

corresponding residual gradients are updated; T_{system} and $T_{compression}$ are calculated for each $(\mathbf{cf}, \delta)$

- **IF $\delta_c \geq \epsilon$ THEN: aggregate compressed gradients $g_c^{(i)}$**
- **IF $\delta_c < \epsilon$ AND $\delta_{min} \geq \epsilon$ THEN: aggregate compressed gradients $g_{min}^{(i)}$**
- **IF $\delta_c < \epsilon$ AND $\delta_{min} < \epsilon$ THEN: aggregate original gradients $g_o^{(i)}$**

GraVAC's Adaptive Compression

- Based on the exploration space and compression step-size, all candidate CFs are evaluated

$$\text{IF } \omega \geq \left|\frac{\delta_{\mathbf{min}} - \delta_{\mathbf{c}}}{\delta_{\mathbf{min}}}\right| \quad \textbf{THEN}\text{: } \textbf{scale up minimum CF as } \mathbf{cf_{min}} = \mathbf{cf_{min}} \cdot \mathbf{c_s}$$

- Once all candidate CFs are evaluated, choose one where $T_{compression}$ saturates

- Compression **scales-up** by increasing $\mathbf{cf_{min}}$ based on $\mathbf{c_s}$ and ω

- Compression **scales-down** according to threshold ϵ

Experimental Evaluation

- GraVAC implemented atop PyTorch 1.10.1 and torch.distributed module
- Evaluated on image and text datasets across 3 popular DL models: **ResNet101**, **VGG16** and **LSTM**
- Deployed over 32 V100 GPUs on the Google Cloud Platform (*n1-standard-8* VMs)
- Compared with static compression techniques like **Top-*k***, **DGC**, **Redsync** and **Random-*k***

Multi-level compression scaling

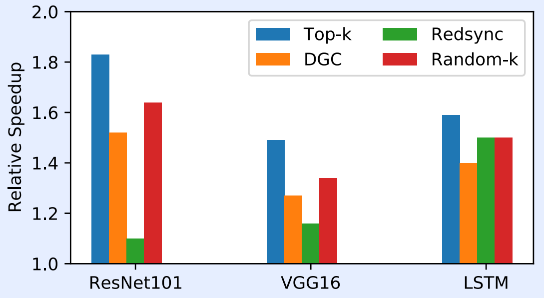
- CFs evaluated in the range $[\mathbf{cf_{min}}, \mathbf{cf_{max}}]$ in steps of $\mathbf{c_s}$
- Compressing original gradients (i.e., $\mathbf{g_o^{(i)}}$) twice can incurs additional compression overhead (i.e., to $\mathbf{g_{min}^{(i)}}$ and $\mathbf{g_c^{(i)}}$)

$$X_1 = C(c_1, X)$$
$$X_2 = C(c_2, X) \mid c_2 > c_1 \text{ and } |X_2| < |X_1| < |X|$$

- **Multi-level compression reduces this overhead by avoiding computation on massive tensors twice!**

$$X_1 = C(c_1, X)$$
$$X_2' = C(c_2', X_1) \ : \ c_2' = \frac{c_2}{c_1} \text{ and } |X_2| = |X_2'|$$

Multi-level (MTL) compression speedup for 10-1000x

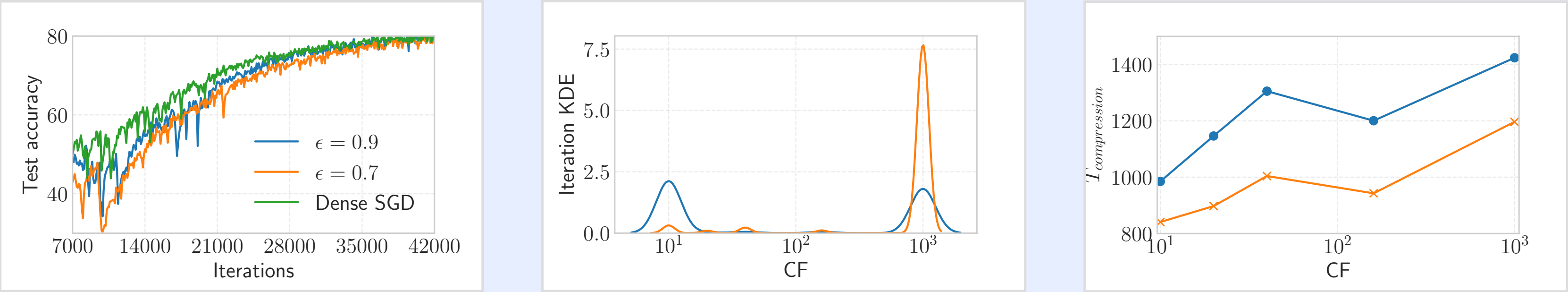


MTL 1.1-1.83x faster than direct compression!

Results

- Train models with CF space [10x, 1000x], w=500 steps, ω=1%, ϵ values 0.7 and 0.9 and c_s increased exponentially

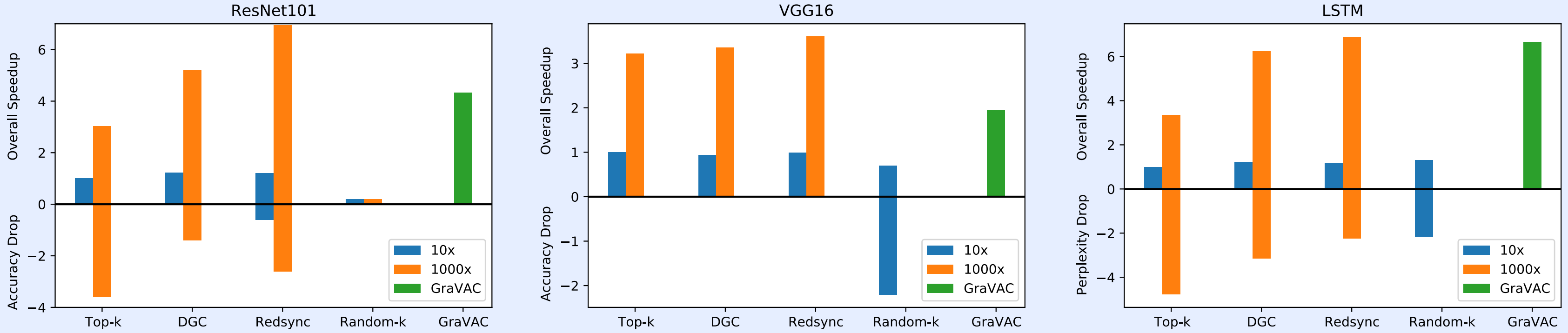
ResNet101



- **Compared to dense-SGD, GraVAC reduces communication volume by 19-163x and achieves the same final accuracy!**
- **VGG16 on GraVAC reduce communication volume by 13-80x; On LSTM by 279-289x**

GraVAC vs. Static compression

- Compared with fixed CFs 10x and 1000x on Top-*k*, DGC, Redsync and Random-*k*
- Overall Speedup reported w.r.t. Top-*k* 10x.
- Accuracy/Perplexity drop reported w.r.t. Dense-SGD.



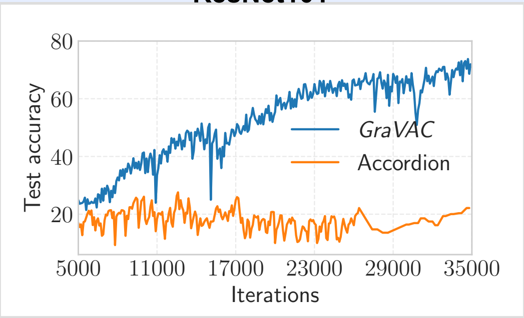
GraVAC and Accordion with Random-*k*

- Despite its low compression overhead, Random-k fails to converge in many cases
- We compare GraVAC with Accordion; both using Random-k compression under the hood

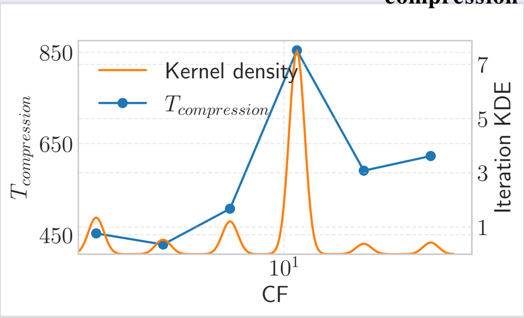
exploration space set to [1.5x, 1000x], w = 2000, ϵ = 0.7, $\mathbf{c_s}$ = 2

Accordion changes CF based on critical regions in training; GraVAC looks at how much information is lost via compression and makes trade-offs between system throughput and accurate gradient representation

ResNet101



GraVAC CF distribution and $\mathbf{T_{compression}}$



GraVAC vs. Accordion

Model	Method	Floats sent	Comm. sav.	Time sav.
ResNet101	Accordion	4.17×10^{11}	1×	1×
	GraVAC	$\mathbf{9.38 \times 10^{9}}$	**44.5×**	**1.94×**
VGG16	Accordion	3.83×10^{11}	1×	1×
	GraVAC	$\mathbf{1.7 \times 10^{10}}$	**22.4×**	**5.63×**
LSTM	Accordion	4.2×10^{11}	1×	1×
	GraVAC	$\mathbf{4 \times 10^{9}}$	**104.2×**	**2.06×**

Related work

- **Gradient noise**: Johnson et al. (AdaScale), Luo et al. (KungFu), Aurick et al. (Pollux), Tyagi et al. (Scavenger)

- **Gradient compression**: Fang et al. (Accelerating Distributed Deep Learning Training with Gradient Compression), Lin et al. (Deep Gradient Compression: Reducing the Communication Bandwidth for Distributed Training), Stitch et al. (Sparsified SGD with Memory)

- **Early phase/Critical region in DNN training**: Jonathan et al. (The Early Phase of Neural Network Training), Alessandro et al. (Critical Learning Periods in Deep Neural Networks)

- **Adaptive gradient compression**: Aggarwal et al. (Accordion: Adaptive Gradient Communication via Critical Learning Regime Identification)

Conclusion

- **Compression gain** helps measure the relative information loss due to compression
- **Compression throughput** works as an effective heuristic to balance the parallel gains of lossy compression and statistical inefficiency of losing gradient information
- GraVAC converges **1.95 - 6.7x** faster than a static CF, while achieving the same convergence as dense-SGD
- **Future directions**:
    - GraVAC on large language models
    - Adaptive compression in model-parallelism
    - Upstream-downstream adaptive compression with Parameter servers in Federated Learning

Thank you!

GraVAC: Adaptive Compression for Communication-Efficient Distributed DL Training