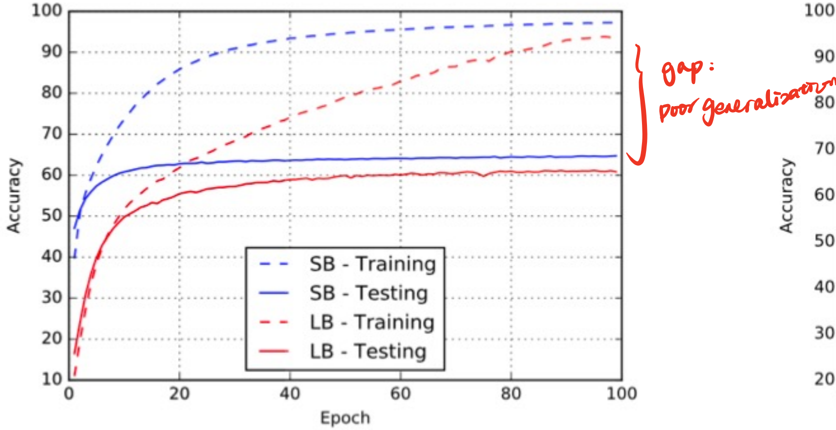
Extension

Accurate, Large Mini-batch SGD

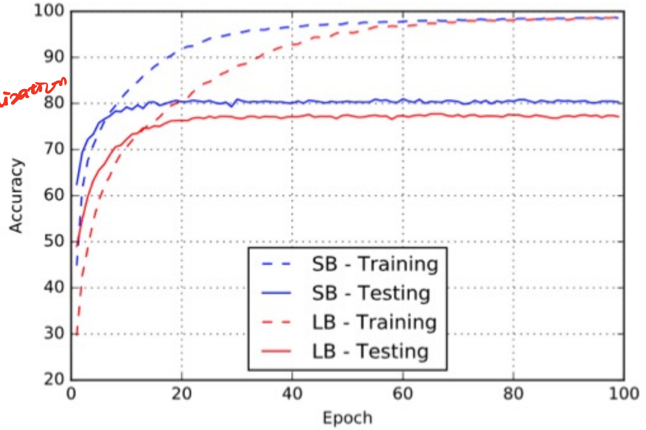
- **Motivation** of scaling up deep learning:
  - Larger datasets and network gives improvement but longer training time.
  - We want to scale up and train faster.
    - ResNet50 on P100/caffe2:
      1GPU/10d-> 8GPUs/29h -> 256GPUs/1h
  - Train visual models on internet-scale data
- Other **Motivations**
  Generalize to object detection and segmentations

Generalization difficulty

effective batch size = batch size * # of workers



(a) Network F_2



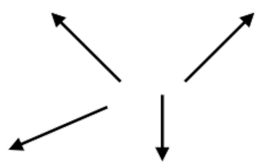
(b) Network C_1

Figure 2: Training and testing accuracy for SB and LB methods as a function of epochs.

Difficulty

- poor generalization (at the end of training)
- optimization difficulty (at the beginning of training)

too many gpus → calculated gradient may not be consistant



Method

- **Method**: (for Distributed Synchronous SGD)
  - Gradient aggregation
  - Learning rate linear scaling + warmup
  - Some tricks to overcome optimization difficulty

**Accurate, Large Minibatch SGD:
Training ImageNet in 1 Hour**

Priya Goyal Piotr Dollár Ross Girshick Pieter Noordhuis
Lukasz Wesolowski Aapo Kyrola Andrew Tulloch Yangqing Jia Kaiming He

Facebook

Distributed Synchronous SGD

$$l(x, w) = \frac{\lambda}{2}\|w\|^2 + \varepsilon(x, w)$$

$$\frac{1}{n}\sum_{x\in\mathcal{B}} \nabla l(x, w) = \lambda w + \frac{1}{n}\sum_{x\in\mathcal{B}} \nabla\mathcal{E}(x, w)$$

grad on local batch i

$$\frac{1}{kn}\sum_{j<k}\sum_{x\in\mathcal{B}_j} \nabla l(x, w_t)$$

grad := grad aggregation

$$\hat{w}_{t+1} = w_t - \hat{\eta}\frac{1}{kn}\sum_{j<k}\sum_{x\in\mathcal{B}_j} \nabla l(x, w_t)$$

weight := weight + f(grad)

Large Minibatch SGD

$$L(w) = \frac{1}{|X|} \sum_{x \in X} l(x, w)$$

$$w_{t+1} = w_t - \eta \frac{1}{n} \sum_{x \in \mathcal{B}} \nabla l(x, w_t)$$

Large Minibatch SGD

Why are we interested in large minibatch SGD?

- The larger mini-batches, the higher per-worker workload, the lower the relative communication overhead (or easier to hide communication overhead) and the easier to scale up.
- We want to use large mini-batches in place of small mini-batches.
- However, using large mini-batches will sacrifice model accuracy in recent literature or simply won't converge.

Learning rates for large minibatch

When the minibatch size is multiplied by k, multiply the learning rate by k.

minibatch

k iterations
update

- k iterations:

$$w_{t+k} = w_t - \eta \frac{1}{n} \sum_{j<k} \boxed{\sum_{x \in \mathcal{B}_j} \nabla l(x, w_{t+j})}$$

adjust
the step size

- single iteration:

$$\hat{w}_{t+1} = w_t - \hat{\eta} \frac{1}{kn} \boxed{\sum_{j<k} \sum_{x \in \mathcal{B}_j} \nabla l(x, w_t)}$$

Learning rate warmup

- Constant warmup

- Gradual warmup

	k	n	kn	η	top-1 error (%)
baseline (single server)	8	32	256	0.1	23.60 ±0.12
no warmup, Figure 2a	256	32	8k	3.2	24.84 ±0.37
constant warmup, Figure 2b	256	32	8k	3.2	25.88 ±0.56
gradual warmup, Figure 2c	256	32	8k	3.2	23.74 ±0.09

Momentum correction

Method 1

$$\begin{cases} u_{t+1} = m u_t + \frac{1}{n} \sum_{x \in \mathcal{B}} \nabla l(x, w_t) \\ w_{t+1} = w_t - \eta u_{t+1}. \end{cases}$$

equivalent

Substituting v_t for ηu_t in (9) yields:

Method 2

$$\begin{cases} v_{t+1} = m v_t + \eta \frac{1}{n} \sum_{x \in \mathcal{B}} \nabla l(x, w_t) \\ w_{t+1} = w_t - v_{t+1}. \end{cases}$$

be careful with learning rate scale issue

Momentum correction

$$
\begin{aligned}
w_{t+1} &= w_t - \eta_{t+1} u_{t+1} \\
&= w_t - \eta_{t+1}\left(m \boxed{u_t} + \frac{1}{n}\sum \nabla l(x, w_t)\right) \\
&= w_t - \eta_{t+1}\left(m \boxed{\frac{v_t}{\eta_t}} + \frac{1}{n}\sum \nabla l(x, w_t)\right) \\
&= w_t - m\frac{\eta_{t+1}}{\eta_t} v_t - \eta_{t+1}\frac{1}{n}\sum \nabla l(x, w_t)
\end{aligned}
$$

if learning rate is the same, no big issue

otherwise correct

So the correct v_{t+1} should be

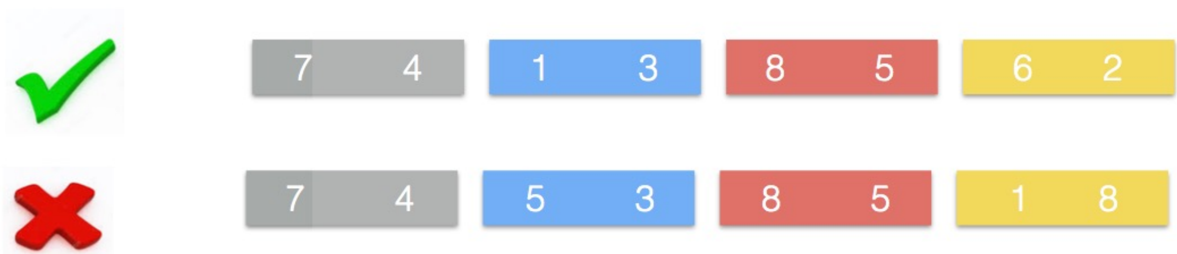
$$
v_{t+1} = m\frac{\eta_{t+1}}{\eta_t} v_t + \eta_{t+1}\frac{1}{n}\sum \nabla l(x, w_t)
$$

21

Data shuffling

Single-worker data shuffling:

4-worker data shuffling:



Weight decay

("regularization")

$$w_{t+1} = w_t - \eta \frac{1}{n} \sum_{x \in \mathcal{B}} \nabla l(x, w_t)$$

$$l(x, w) = \frac{\lambda}{2} \|w\|^2 + \varepsilon(x, w)$$

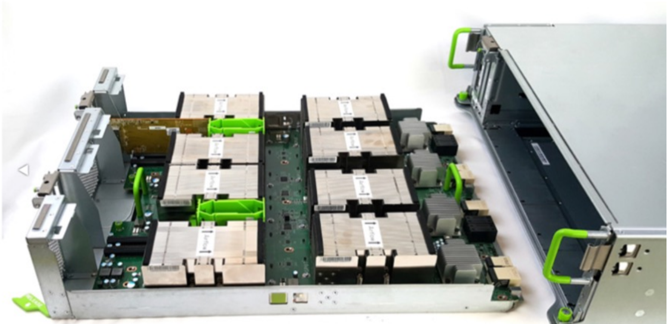
only modify this term → since large minibatch are pool worker

$$w_{t+1} = w_t - \eta \lambda w_t - \eta \frac{1}{n} \sum_{x \in \mathcal{B}} \nabla \varepsilon(x, w_t)$$

Implementations

Gradient Aggregation

- within a server:
  - if data>256kb, use NCCL
  - else, GPU->host + reduction
- between servers:
  - recursive halving and doubling algorithm
- Non-power-of-two servers:
  - binary blocks algorithm



credit: https://code.facebook.com/posts/1835166200089399/introducing-big-basin

Intel-based
8 P100 GPUs with NVLink
3.2T NVMe SSDs
Mellanox 50G Ethernet

Results

Minibatch size vs. error

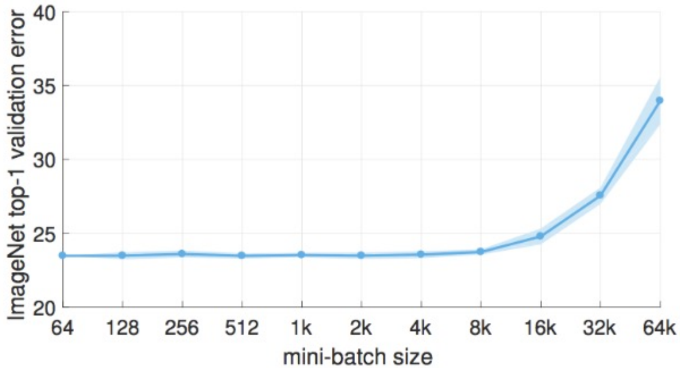
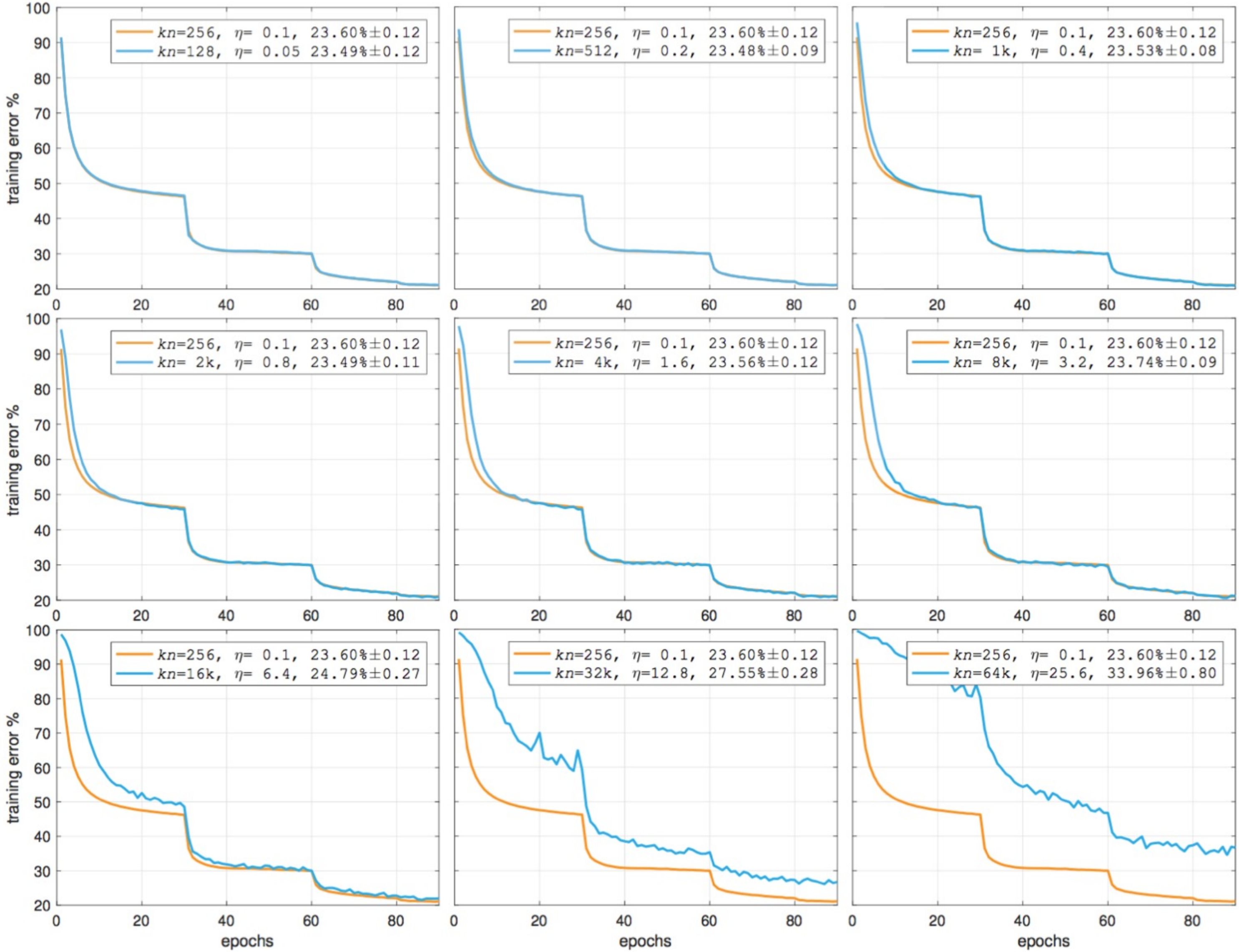


Figure 1. **ImageNet top-1 validation error *vs*. minibatch size.**



Goyal et al. Accurate, Large Minibatch SGD: Training ImageNet in 1 hour