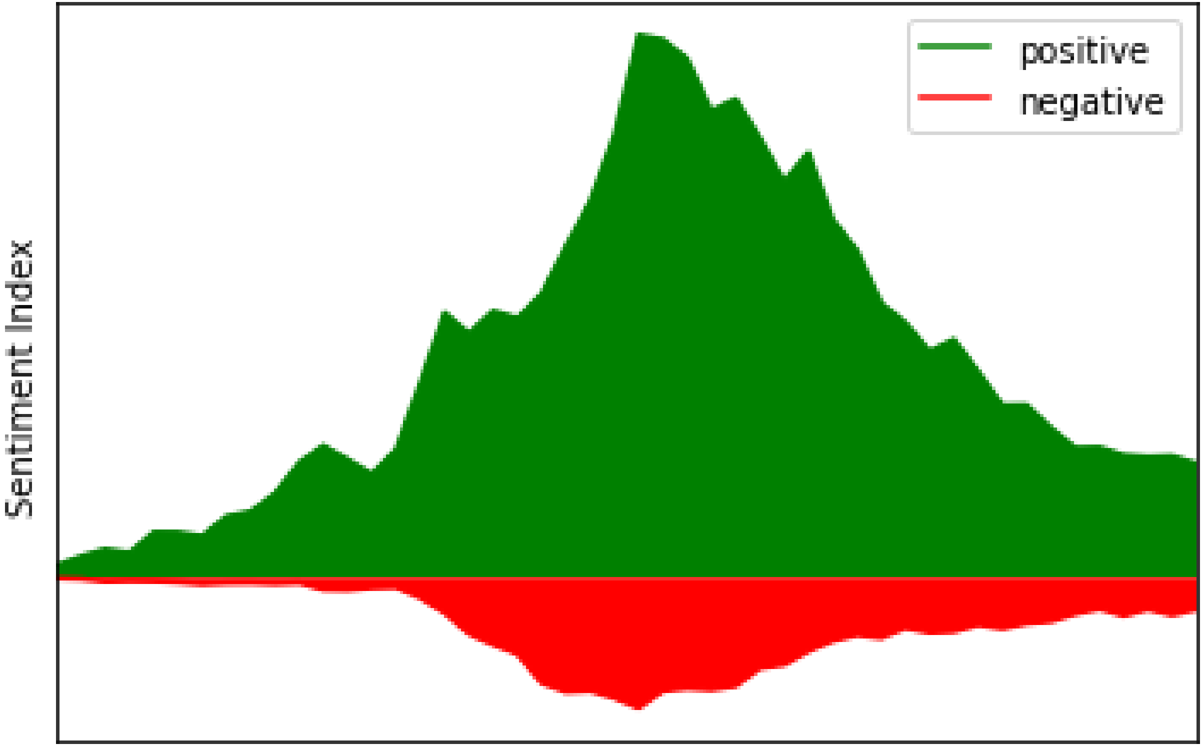
Large Scale Language Modeling: Converging on 40GB of Text in Four Hours

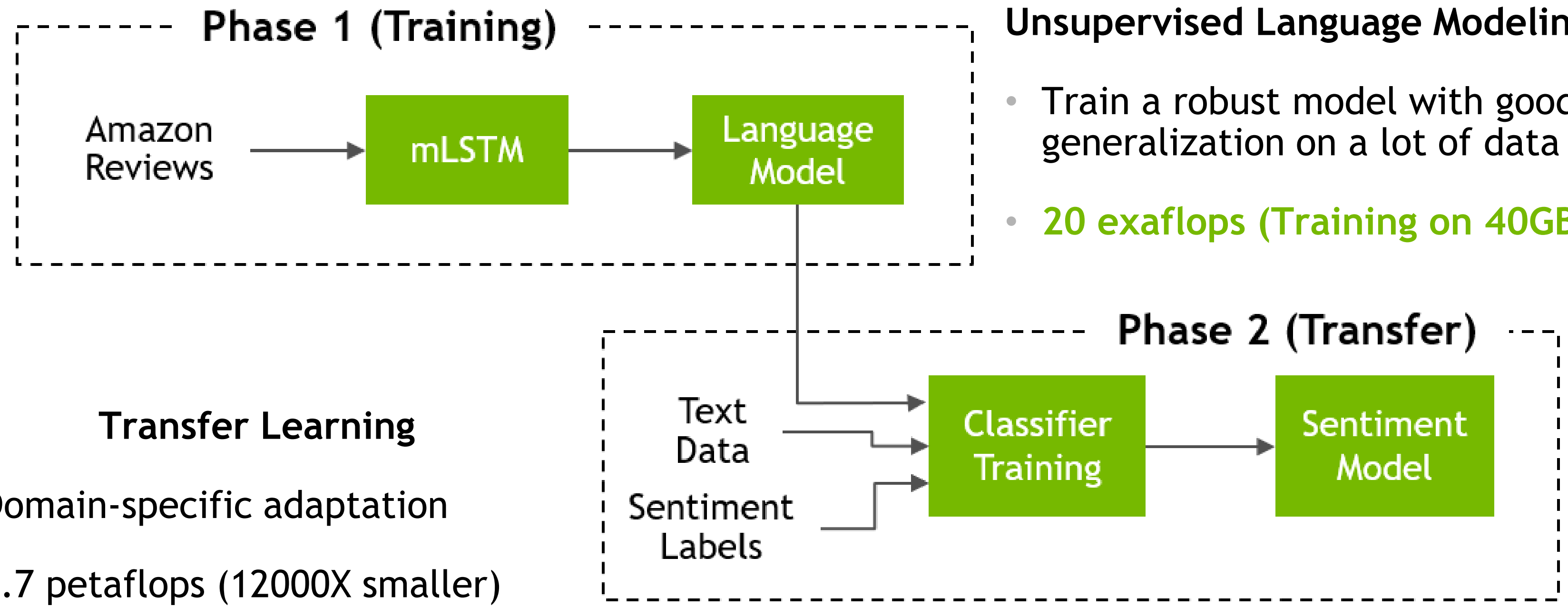
Raul Puri, Robert Kirby, Nikolai Yakovenko, Bryan Catanzaro

SENTIMENT ANALYSIS

Making sense of text data



LANGUAGE MODEL PRETRAINING & TRANSFER



Unsupervised Language Modeling

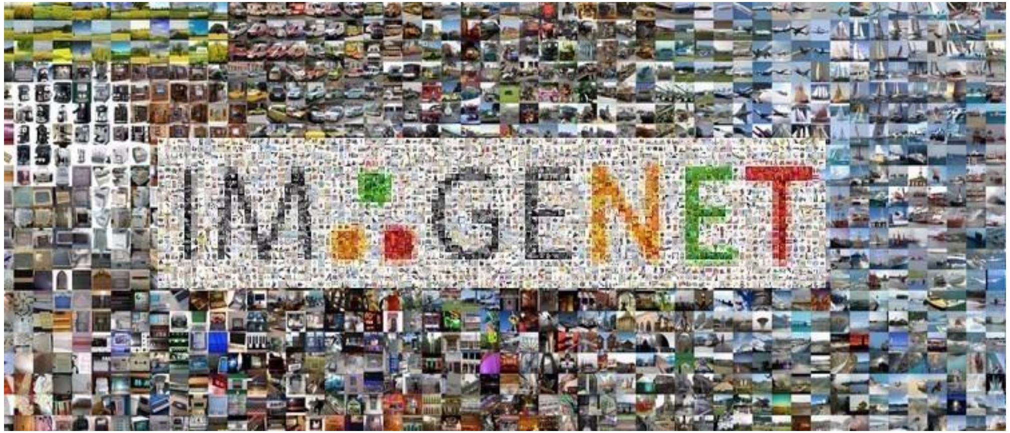
- Train a robust model with good generalization on a lot of data
- 20 exaflops (Training on 40GB)

Transfer Learning

- Domain-specific adaptation
- 1.7 petaflops (12000X smaller)

RELATED WORK

- Pretraining + transfer has been used in the computer vision community



Source

transfer

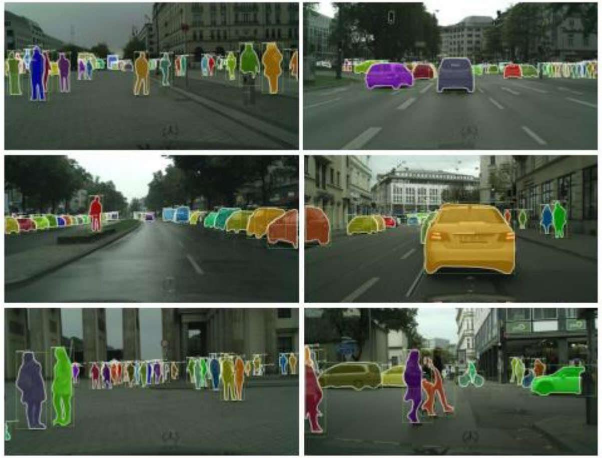


Figure 8. Mask R-CNN results on Cityscapes test (32.0 AP).

Source

RELATED WORK

- Language Model (LM) pre training has moved from training word vectors, to training entire models
- Examples:
    - Semi-supervised Sequence Learning
    - context2vec
    - CoVe
    - ELMo

RELATED WORK

- Proliferation of recent work with LM pretraining + transfer
    - Even more tasks and models
    - State of the Art results
- Examples:
    - Learning to Generate Reviews and Discovering Sentiment
    - Improving Language Understanding by Generative Pre-training
    - Universal Language Model Fine-Tuning for Text classification
    - Learning general purpose distributed sentence representations via large scale multi-task learning
    - Universal Sentence Encoder

DEEP LEARNING AT SCALE

- One central problem cited in these works is training scale
- How to scale DL+NLP training is not investigated to same extent as DL+CV
- Addressing scale concerns is necessary to advance state of the art NLP research

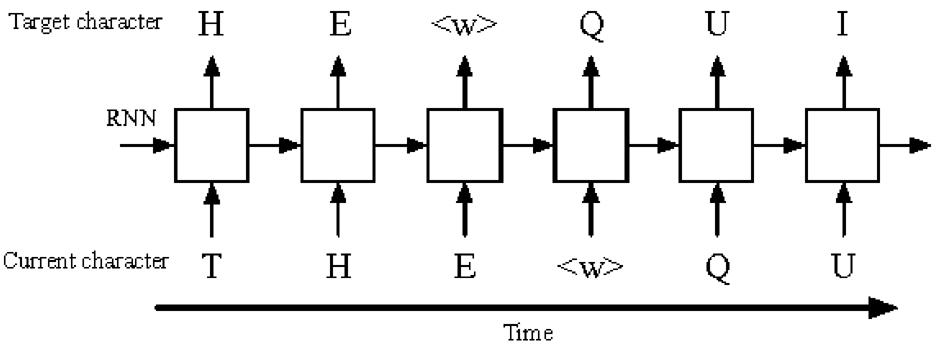
SCALING PRETRAINING

- FP16 tensorcores
- Multi-GPU/Multi-node training
- Large batch training/Distributed Data Parallelism
    - Large batch learning rate schedule

UNSUPERVISED TRAINING

- Train a 4096-d multiplicative LSTM (mLSTM) recurrent neural network (RNN) via next character prediction
  - Model learns dynamics of language
  - NO LABELS NEEDED - Label is next character

- 40GB of sentiment-filled Amazon Review data



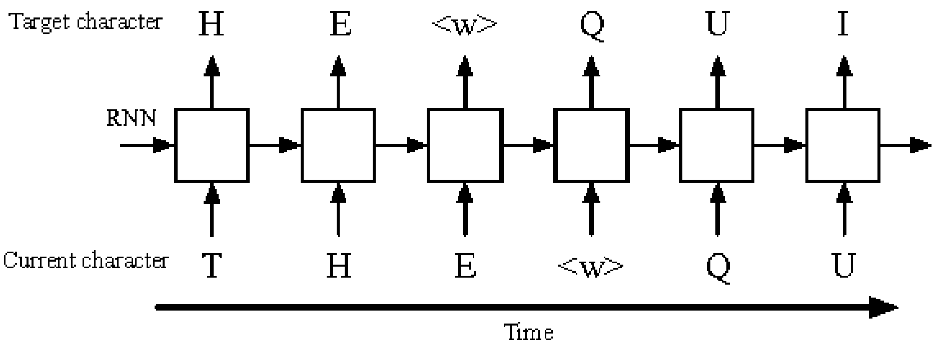
source

Sample Reviews

Shadows was an amazing book that caught my imagination instantly! It had love, brutality, adventure, and suspense that captivates your mind throughout the whole book.
the hooks were not chipped shipping was really fast nothing was broken all hooks were in package as described with all the sizes A+++++ thank you
Love this feeder. Heavy duty & capacity. Best feature is the large varmint guard. Definitely use a small lock or securing device on the battery housing latch. I gave 4 stars because several bolts were missing. Check contents b4 beginning.
The mp3 comes in Chinese!!! I DON'T KNOW THAT LANGUAGE, I AM ORDERING FOM USA. I DON'T UNDERSTAND ANYTHING AND I AM NOT ABLE TO CHANGE IT!!

UNSUPERVISED TRAINING

- Amazon is one of the largest good quality datasets
- Prior implementations from Radford et al trained with:
  - Weight Normalization
  - Adam @ Batch 128 with 5e-4 learning rate decaying to 0 over 1 epoch
  - TOOK 1 MONTH



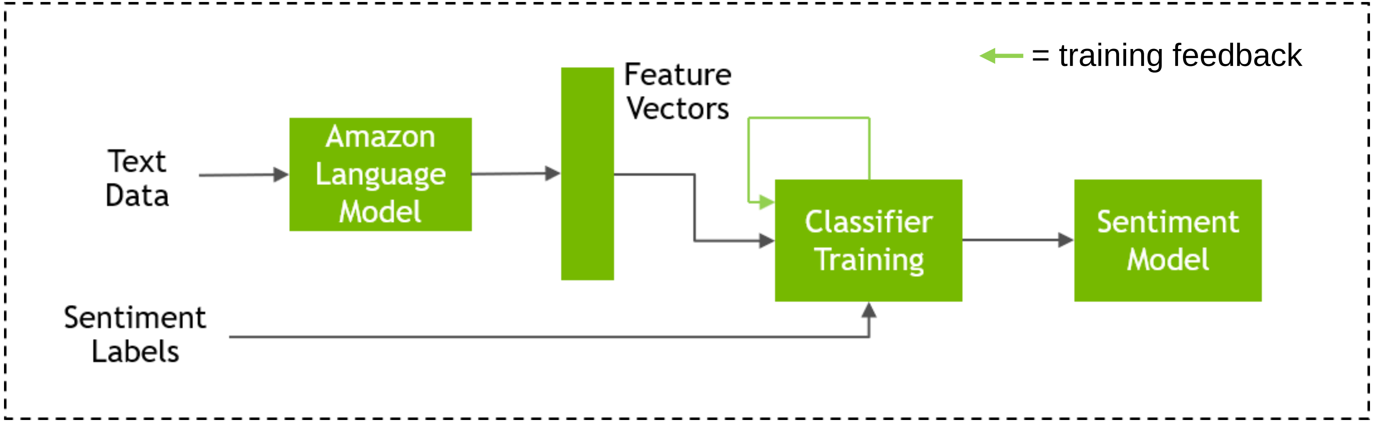
source

Sample Reviews

Shadows was an amazing book that caught my imagination instantly! It had love, brutality, adventure, and suspense that captivates your mind throughout the whole book.
the hooks were not chipped shipping was really fast nothing was broken all hooks were in package as described with all the sizes A+++++ thank you
Love this feeder. Heavy duty & capacity. Best feature is the large varmint guard. Definitely use a small lock or securing device on the battery housing latch. I gave 4 stars because several bolts were missing. Check contents b4 beginning.
The mp3 comes in Chinese!!! I DON'T KNOW THAT LANGUAGE, I AM ORDERING FOM USA. I DON'T UNDERSTAND ANYTHING AND I AM NOT ABLE TO CHANGE IT!!

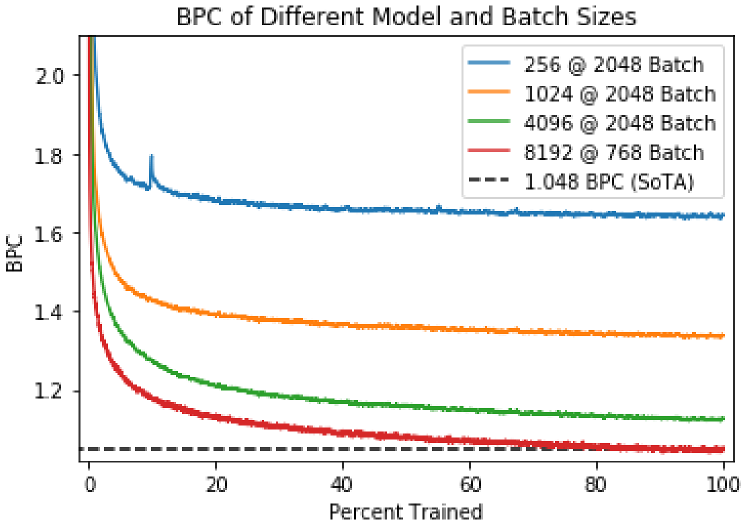
TRANSFER LEARNING

1. Initialize model with weights from pretraining
2. Model is used to featurize bodies of text (IMDB Reviews/Stanford Sentiment Treebank)
3. Binary Sentiment Classifier is trained on text features
4. Output Model: language model base + classifier on top



UNSUPERVISED TRAINING – LARGE MODELS

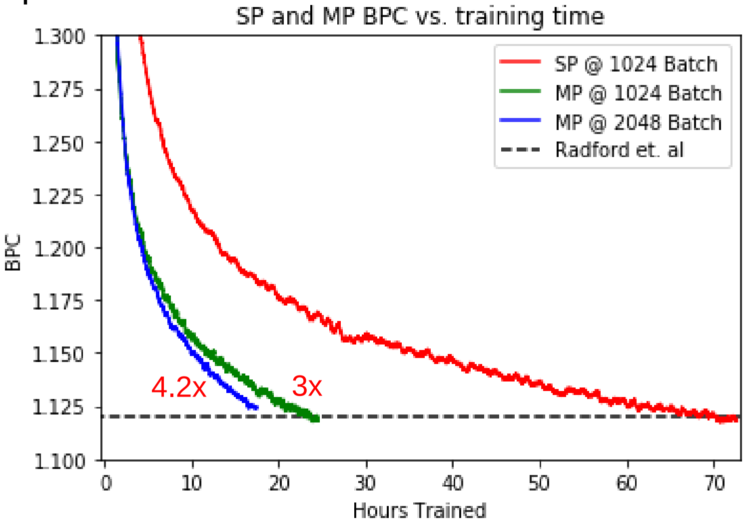
- Pretraining + transfer works with different model sizes
- Bigger better language model = better transfer
- Pretraining large models is expensive
- Scaling training is necessary for practicality



Hidden Size	FLOPS LM	FLOPS Transfer	BPC	SST	IMDB
256	1.14e17	3.19e12	1.541	53.2	62.2
1024	1.35e18	1.14e14	1.263	81.8	76.2
4096	**2.01e19**	**1.67e15**	**1.073**	**91.5**	**92.8**
8192	7.91e19	6.62e15	1.036	93.8	94.8

SCALING TRAINING – VOLTA TENSOR CORES

- 4.2x faster training in mixed precision on tensorcores
  - Reduced memory footprint allows for 2x batch size

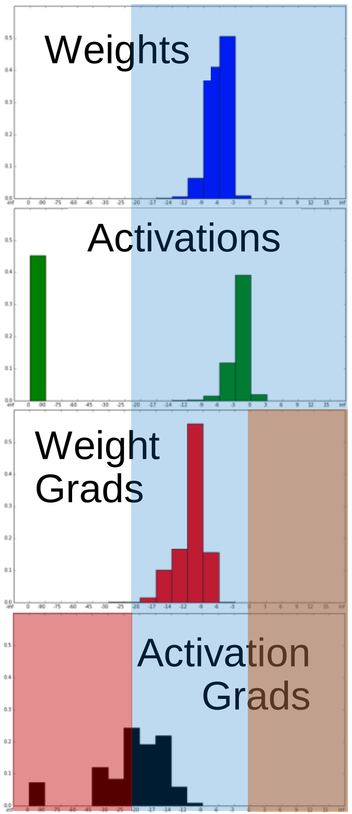


SP = Single Precision
MP = Mixed Precision

Type	Batch	LR	Time	BPC	SST	IMDB
SP	**128**	**5e-4**	**1 month**	**1.12**	**91.9**	**92.8**
SP	1024	1.2e-3	73.8 hr	1.104	90.8	92.5
MP	1024	1.2e-3	24.2 hr	1.108	91.5	91.7
MP	2048	2e-3	17.4 hr	1.117	90.2	91.9

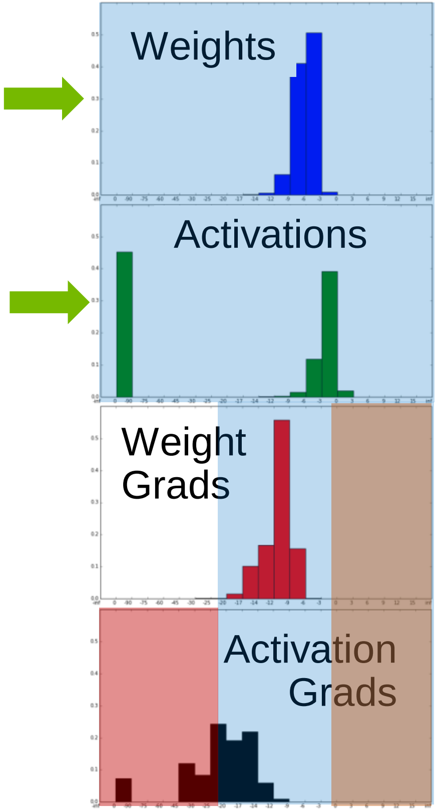
FP16 UNDERFLOW

- Range representable in FP16: ~40 powers of 2
- Gradients are small:
    - Some lost to zero, while ~15 powers of 2 unused



FP16 UNDERFLOW

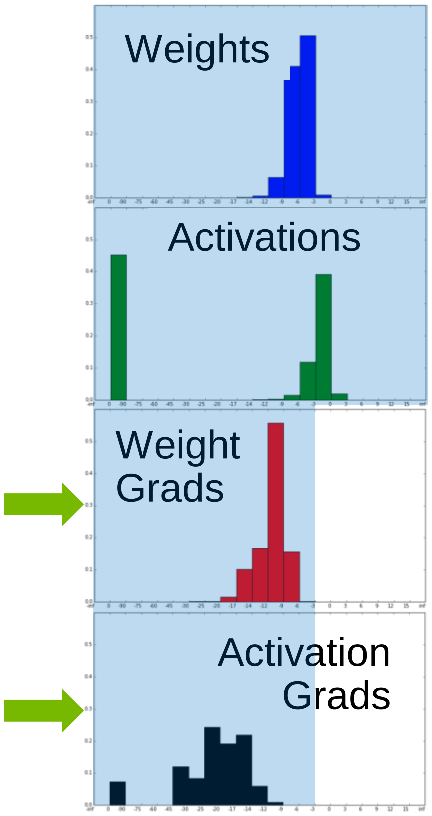
- Range representable in FP16: ~40 powers of 2
- Gradients are small:
    - Some lost to zero, while ~15 powers of 2 unused
- Unsafe Operations done in FP32 (ie. softmax loss)
- Master Copy of Parameters in FP32 to accumulate gradient updates



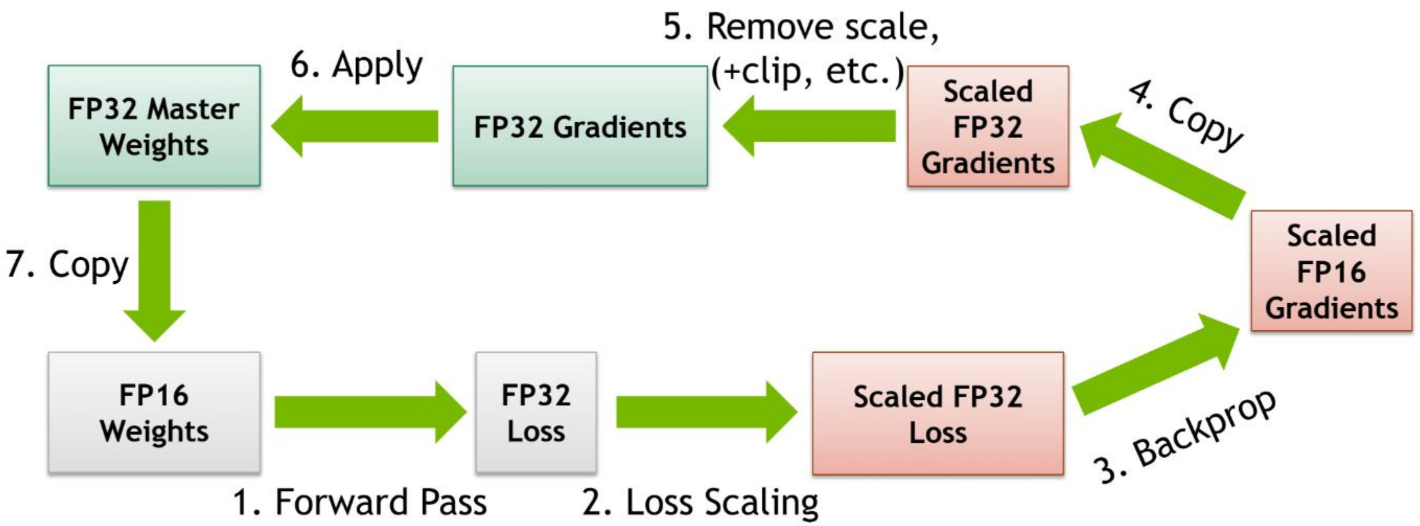
LOSS SCALING

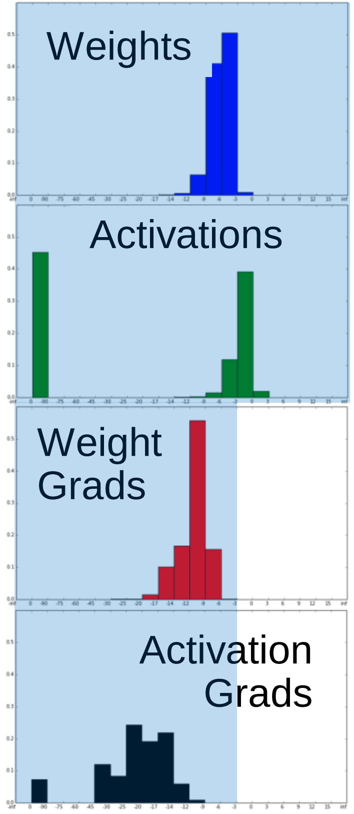
1. Multiply loss by a constant S
2. All gradients scaled up by S (chain rule)
3. Unscale weight gradient (in FP32) before weight update

Loss scale chosen by automatic loss scaling



FP16 TRAINING





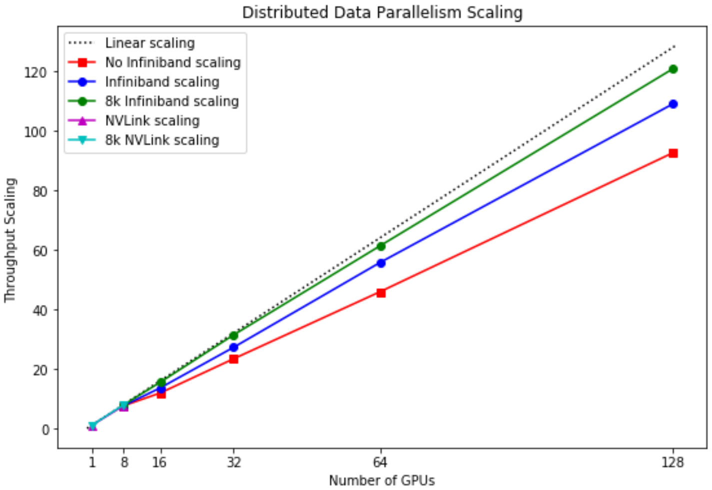
MULTI-GPU TRAINING

- Leverage multiple GPUs to accelerate training
- Data Parallelism vs. Model Parallelism
    - Large Batch Distributed Data Parallelism is model agnostic
        - Great for RNNs

DISTRIBUTED DATA PARALLEL SCALING

- Near linear throughput scaling
    - NVLink+NCCL2 within node
    - Infiniband across nodes

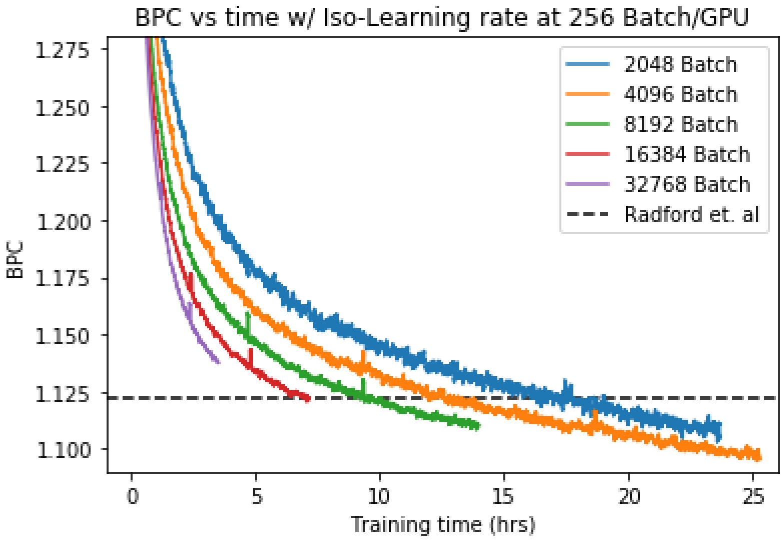
- Larger, compute-dominant models scale even better



GPUs	w/o I.band		w/ I.band		8192-d + I.band	
	s/iter	speed	s/iter	speed	s/iter	speed
1	.81	1x	.81	1x	2.01	1x
8	.85	7.6x	.85	7.6x	2.02	7.9x
16	1.09	14.3x	.91	13.6x	2.08	15.5x
32	1.11	23.4x	.91	27.2x	2.05	31.4x
64	1.13	55.7x	.93	55.7x	2.10	61.3x
128	1.12	92.6x	.91	109x	2.13	120.8x

LARGE BATCH TRAINING

- Train with 32k batch size on 128 GPUs
- Converges with reasonable transfer accuracy in 3.5 hours
    - 3e-3 Learning Rate
    - Decay to 0 over 100k steps or 3 epochs
        - (whichever comes first)
    - No shuffling in between epochs



Batch	GPU	Iters	Ep	hrs	BPC	SST	IMDB
2048	8	100k	1.4	23.7	1.102	90.6	92.1
4096	16	100k	2.7	25.3	1.090	90.6	92.7
8192	32	55k	3.0	14.0	1.104	91.2	92.3
16384	64	28k	3.0	7.1	1.116	90.3	92.3
32768	128	14k	3.0	3.5	1.132	90.1	90.4

LARGE BATCH TRAINING - LR SCALING

- Modifying the learning rate is necessary to converge with Large Batch Training
- Several rules proposed in recent literature
  a. Linear Scaling: $\epsilon \propto B$ [17,39,40]
  b. Square Root Scaling: $\epsilon \propto B^{1/2}$ [41]
- Both assume that N >> B (ϵ=learning rate, N=dataset size, B=batch size)
- Designed with SGD in mind not Adam

Example LR Scaling

Batch	Linear	Square Root
128	5e-4	5e-4
256	1e-3	7.2e-4
512	2e-3	1e-3

17: Accurate, Large Minibatch SGD: Training ImageNet in 1 Hour
39: A Bayesian Perspective on Generalization and Stochastic Gradient Descent
40: Don't decay the learning rate, increase the batch size
41: Train longer, generalize better: closing the generalization gap in large batch training of neural networks

LARGE BATCH TRAINING - LR SCHEDULES

- Linear LR scaling diverges easily
  - Square root does better, but also diverges
- Some scaling is necessary
  - Original 5e-4 LR does worse in every setting
  - 3e-3 performed best across multiple settings

Batch	Iters	Rule	LR	BPC	SST	IMDB
2048	72.6k	linear	8e-3	1.280	79.4	77.6
		sqrt	2e-3	1.117	90.2	91.9
		-	5e-4	1.130	89.1	90.8
		-	3e-3	1.110	89.0	92.1
4096	37.3k	linear	1.6e-2	1.275	78.3	77.6
		sqrt	2.8e-3	1.122	89.6	91.0
		-	5e-4	1.146	89.3	90.9
		-	3e-3	1.119	89.2	91.8
8192	18.6k	linear	3.2e-2	1.476	65.4	67.3
		sqrt	4e-3	1.133	89.7	90.8
		-	5e-4	1.175	87.3	89.6
		-	3e-3	1.132	89.5	91.4
16384	9.3k	linear	6.4e-2	Div	-	-
		sqrt	5.8e-3	Div	-	-
		-	5e-4	1.254	85.1	86.4
		-	3e-3	1.162	89.0	90.1
32768	4.6k	linear	1.3e-1	Div	-	-
		sqrt	8e-3	Div	-	-
		-	5e-4	1.380	75.2	74.8
		-	3e-3	1.218	87.1	87.9

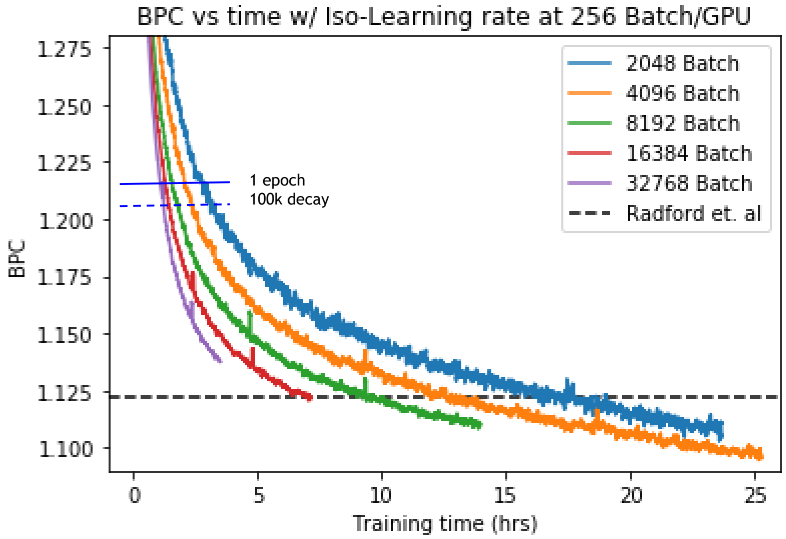
LARGE BATCH TRAINING - LR SCHEDULES

- Linear LR scaling diverges easily
  - Square root does better, but also diverges
- Some scaling is necessary
  - Original 5e-4 LR does worse in every setting
  - 3e-3 performed best across multiple settings
- Similar learning rates in different decay and batch conditions perform differently

Batch	Iters	Rule	LR	BPC	SST	IMDB
2048	72.6k	linear	8e-3	1.280	79.4	77.6
		sqrt	2e-3	1.117	90.2	91.9
		-	5e-4	1.130	89.1	90.8
		-	3e-3	1.110	89.0	92.1
4096	37.3k	linear	1.6e-2	1.275	78.3	77.6
		sqrt	2.8e-3	1.122	89.6	91.0
		-	5e-4	1.146	89.3	90.9
		-	3e-3	1.119	89.2	91.8
8192	18.6k	linear	3.2e-2	1.476	65.4	67.3
		sqrt	4e-3	1.133	89.7	90.8
		-	5e-4	1.175	87.3	89.6
		-	3e-3	1.132	89.5	91.4
16384	9.3k	linear	6.4e-2	Div	-	-
		sqrt	5.8e-3	Div	-	-
		-	5e-4	1.254	85.1	86.4
		-	3e-3	1.162	89.0	90.1
32768	4.6k	linear	1.3e-1	Div	-	-
		sqrt	8e-3	Div	-	-
		-	5e-4	1.380	75.2	74.8
		-	3e-3	1.218	87.1	87.9

LARGE BATCH TRAINING - LR SCHEDULES

- Linear LR scaling diverges easily
    - Square root does better, but also diverges
- Some scaling is necessary
    - Original 5e-4 LR does worse in every setting
    - 3e-3 performed best across multiple settings
- Similar learning rates in different decay and batch conditions perform differently
- Performance after 1 epoch with slower decay is also better



Batch	Iters	Rule	LR	BPC	SST	IMDB
32768	4.6k	linear	1.3e-1	Div	-	-
		sqrt	8e-3	Div	-	-
		-	5e-4	1.380	75.2	74.8
		-	3e-3	1.218	87.1	87.9

DATASET AND BATCH SIZE EFFECTS CONVERGENCE

- Relationship between Batch size, lr, and convergence is complex
    - Factors: dataset size/training length, lr decay, optimizer choice
    - Learning rate scaling alone is not enough

- Even with a large text dataset, convergence with 32k batch is feasible but difficult
    - $N > B$, not $N >> B$
    - Running for more epochs helps

CONCLUSION

- Leveraging today’s compute allows us to accelerate training time from months to hours
- FP16 mixed precision training can be used to converge RNN training faster
- Large batch training can be used to train language models, but choosing a learning rate schedule is complex and dependent on multiple factors
- Accelerating training leads to quicker development of downstream applications

FUTURE WORK

- Longer Training
- Learning rate schedule search
- Larger Models
- Different models & different tasks than sentiment analysis

THANK YOU

- PyTorch Code: https://github.com/NVIDIA/sentiment-discovery
- FP16 + Distributed training interface: https://github.com/NVIDIA/apex
- Paper: https://arxiv.org/abs/1808.01371
- Contact: raulp@nvidia.com