TSPipe: Learn from Teacher Faster with Pipelines

Hwijoon Lim, Yechan Kim, Sukmin Yun, Jinwoo Shin, Dongsu Han
Korea Advanced Institute of Science and Technology (KAIST)

TSPipe accelerates training of Knowledge Distillation (KD) and Self-Supervised Learning (SSL) networks with pipelines.

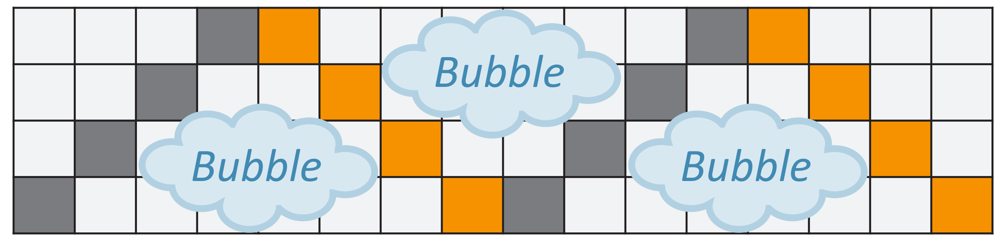
Motivation: Accelerating KD and SSL

Teacher-Student (TS) Framework

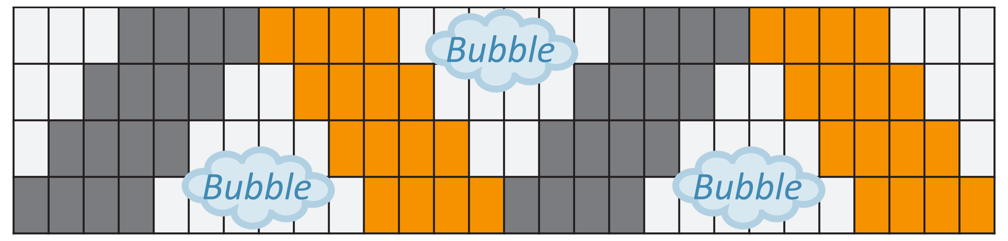
- Teacher-student (TS) framework is commonly adopted in **Knowledge Distillation (KD)**
- Also adopted by many momentum-based **Self-Supervised Learning (SSL)** networks
  - Teacher network ξ_n is slowly updated as an exponential moving average of student θ_n

How can we train large models that do not fit in a single GPU

- Some large models cannot be trained as a whole, even with a cutting-edge GPU
- Model Parallelism **split a model** into multiple partitions and train with multiple GPUs
  → **serious GPU under-utilization** due to the dependency between partitions
- Pipeline Parallelism pipelines computation of each batch for better GPU utilization
  → Approaches that preserve training semantics (e.g. GPipe) **fail to fully utilize GPUs**
  → Approaches that achieve higher utilization **incur overheads** (e.g. memory, accuracy)



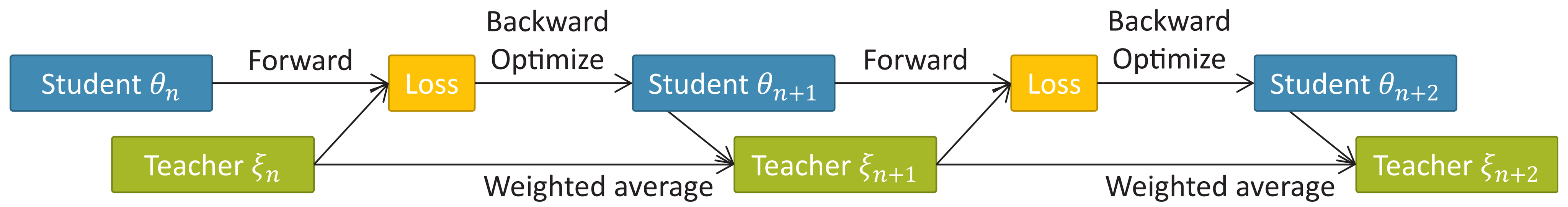
Inter-Layer Model Parallelism



Pipeline Parallelism (GPipe)

Challenge

- **Can we fully schedule the computations** despite the dependency between them?
  - To compute the teacher ξ_{n+1}, we need to wait for student θ_{n+1} to be computed
- **Can we eliminate pipeline bubbles** by inserting computations while GPUs are idle?
  - Reordering computations may require activation stashing for gradient calculation

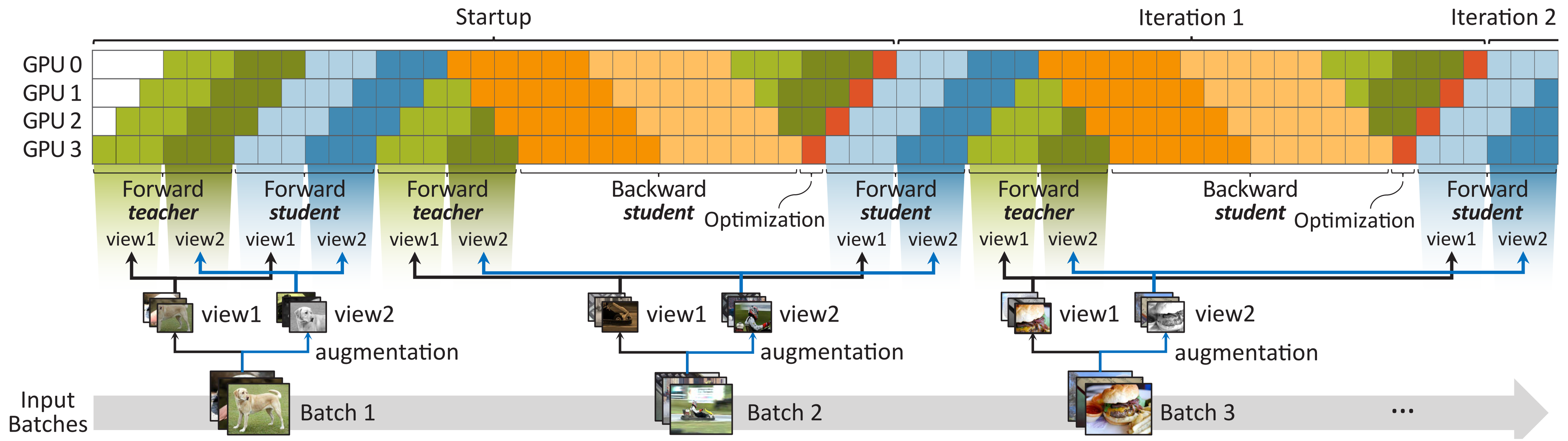


Teacher network **does not need a backward pass**
→ Teacher network's forward pass can be **scheduled more leniently** without activation stashing

How TSPipe works

Key Idea

- **Separate the scheduling** of student and teacher networks
- **Interleave teacher's forward pass** between the computations of student networks



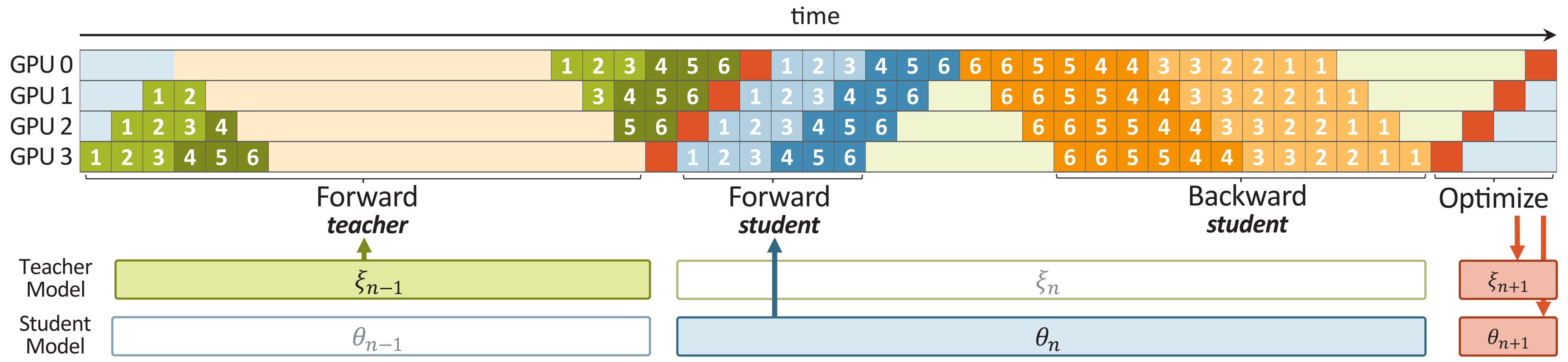
- Use forward pass of the teacher from the previous iteration and forward pass of the student from the current iteration to compute loss
- **Schedules 100% GPU pipeline** without pipeline bubbles and activation stashing

Attaining high model accuracy

- Many existing Pipeline Parallelism schemes **change training schemes** to train faster

$$\theta_{n+1} \leftarrow \text{optimizer}(\theta_n, \nabla \mathcal{L}_{\theta_{n-1}, \xi_{n-1}}, \eta)$$

→ Usually comes with model staleness, which degrades accuracy



- We leverage that teacher network $\xi_{n-1} \approx \xi_n$ (since $\xi_{n+1} \leftarrow \tau\xi_n + (1-\tau)\theta_{n+1}$, where $\tau \approx 1$)
- **Preserve model accuracy** by introducing **asymmetric parameter update** as

$$\theta_{n+1} \leftarrow \text{optimizer}(\theta_n, \nabla_{\theta_n} \mathcal{L}_{\theta_n, \xi_{n-1}}, \eta)$$

where we make only the teacher network stale

Experimental Results

	Method	Architecture	Param.	Training Throughput (Seq/s): Inter-layer MP	GPipe	**TSPipe** (Ours)
KD	Soft Target (Hinton et al., 2015)	ViT-Large / ResNet-101	303 M / 43 M	57.41	136.8	204.4 **(3.56x)**
		ViT-Large / ResNet-152	303 M / 58 M	47.24	126.6	180.7 **(3.82x)**
		ViT-Huge / ResNet-101	631 M / 43 M	35.65	100.6	148.5 **(4.17x)**
		ViT-Huge / ResNet-152	631 M / 58 M	30.30	84.03	141.8 **(4.68x)**
	DistillBERT (Sanh et al., 2019)	BERT-xlarge	1.3 B / 480 M	62.82	113.3	193.4 **(3.08x)**
		BERT-xxlarge	3.9 B / 1.2 B	30.36	75.22	98.82 **(3.25x)**
SSL	BYOL (Grill et al., 2020)	ResNet-18	11 M	346.3	585.1	728.5 **(2.10x)**
		ResNet-50	26 M	102.0	232.0	295.8 **(2.90x)**
		ResNet-101	45 M	71.25	162.7	243.0 **(3.41x)**
		ResNet-152	60 M	53.33	136.9	201.6 **(3.78x)**
	MoCo-v3 (Chen et al., 2021)	ViT-Small	22 M	99.42	259.9	365.7 **(3.68x)**
		ViT-Base	86 M	35.06	106.7	176.6 **(5.04x)**
		ViT-Large	307 M	11.31	33.95	54.70 **(4.84x)**
		ViT-Huge	632 M	5.496	18.71	35.26 **(6.42x)**

Training throughput (seq/s) on 8 V100 GPUs

- Achieve up to **6.42x (with 8 GPUs)** and **12.15x (with 16 GPUs)** higher training throughput than Inter-layer Model Parallelism (MP)
- **Best performance improvement in large models** (MoCo-v3 + ViT-Huge)
  → Comes from higher utilization of internal computing resources in GPUs

Effectiveness of Asymmetric Parameter Update

Dataset	Vanilla Top1	Vanilla Top5	TSPipe Top1	TSPipe Top5
STL10 (Coates et al., 2011)	81.73 ± 0.27	99.41 ± 0.06	81.75 ± 0.32 (+0.02)	99.40 ± 0.03 (-0.01)
CIFAR10 (Krizhevsky et al., 2009)	74.76 ± 0.34	98.60 ± 0.08	75.24 ± 0.52 (+0.48)	98.73 ± 0.09 (+0.13)
CIFAR100 (Krizhevsky et al., 2009)	48.54 ± 0.34	78.46 ± 0.16	49.79 ± 0.32 (+1.25)	79.22 ± 0.50 (+0.76)
ImageNet100 (Russakovsky et al., 2015)	64.18 ± 0.61	88.12 ± 0.33	64.24 ± 0.23 (+0.06)	88.24 ± 0.22 (+0.12)

Linear Evaluation Accuracy (BYOL with ResNet-18)

- TSPipe **preserves the final model accuracy** without any tradeoffs
- Ablation study shows significant accuracy drops (up to -5.9%p) without asymmetric parameter update