NVIDIA

MIXED PRECISION TRAINING FOR 3D MEDICAL IMAGE ANALYSIS

Dong Yang

August 23, 2020

INTRODUCTION

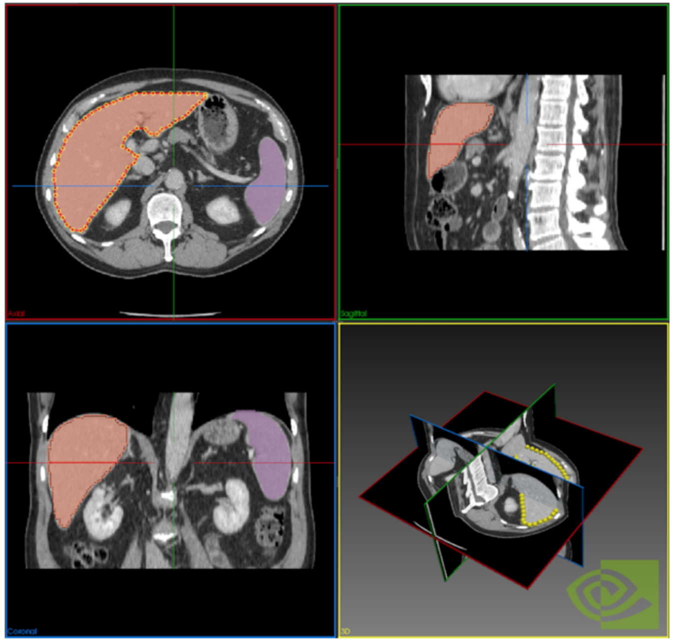
Medical Image Analysis

- What?
  - To gain high-level understanding from medical images (CT, MRI, X-ray, ultrasound, etc.)
- Why?
  - Anatomical understanding, disease diagnosis, treatment planning, and surgery guidance
  - e.g. “COVID-19” screening
- How?
  - Machine learning, deep learning, etc.

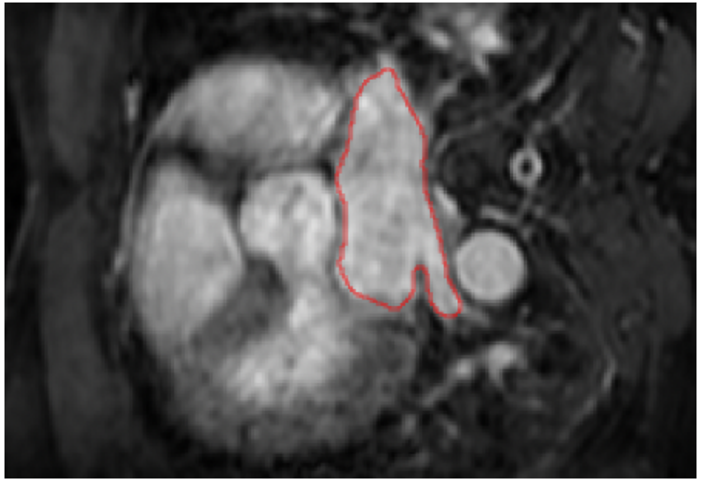
APPLICATION

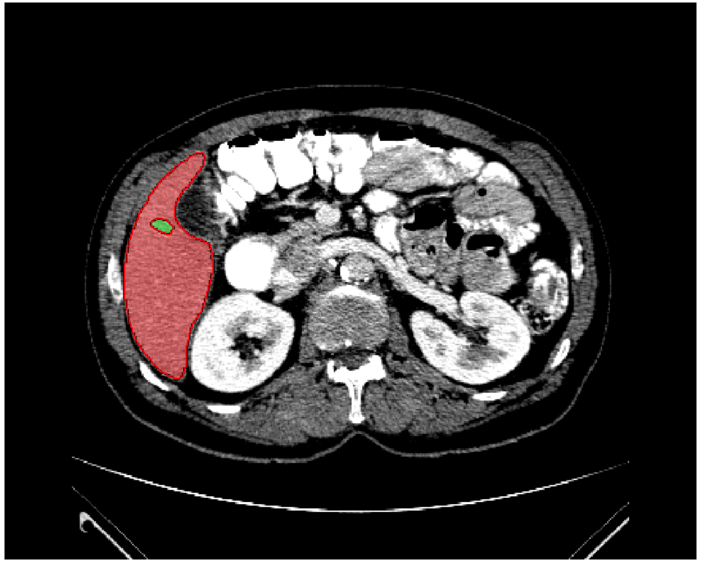
3D Medical Image Segmentation

Given 3D volumes (e.g. CT, MRI) as input, to extract 3D structures of organs or tumors



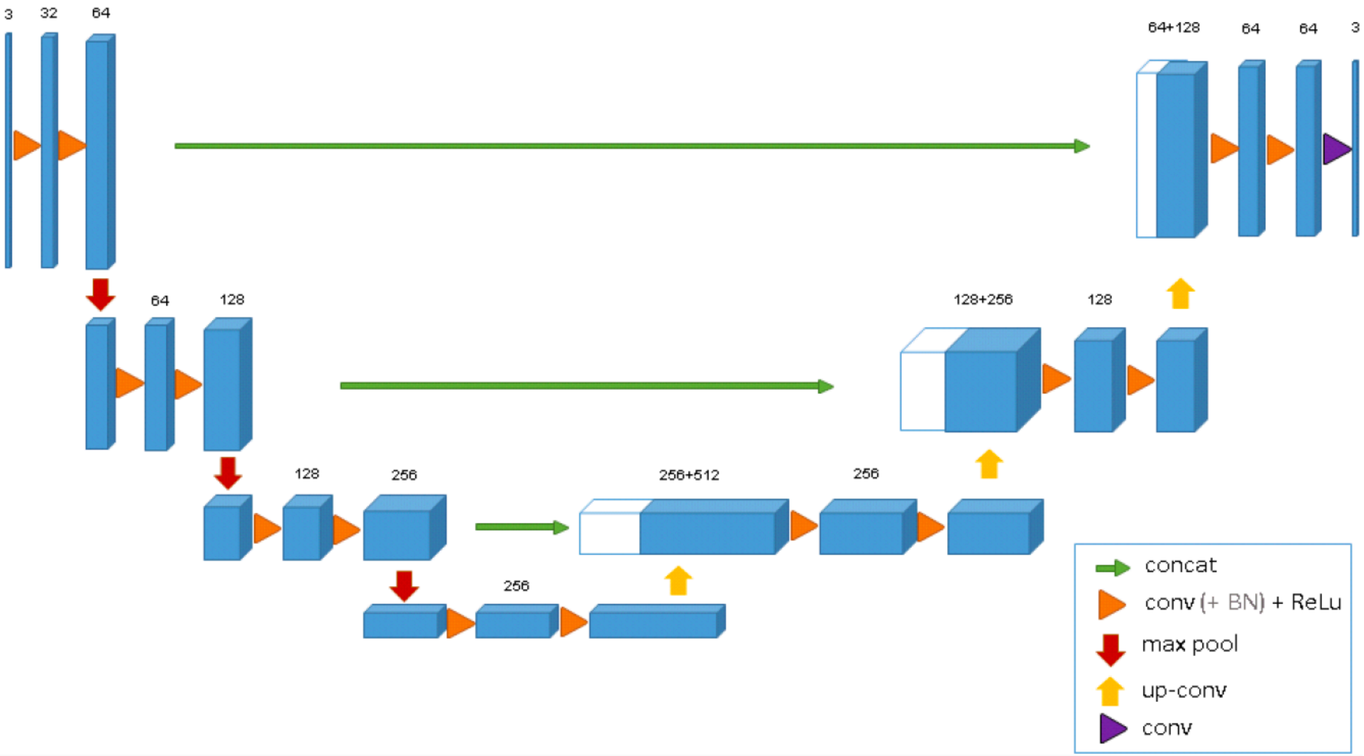
https://devblogs.nvidia.com/annotation-transfer-learning-clara-train/
https://pbs.twimg.com/media/DBozgARUQAAE66r.jpg





BASELINE

3D U-Net Model

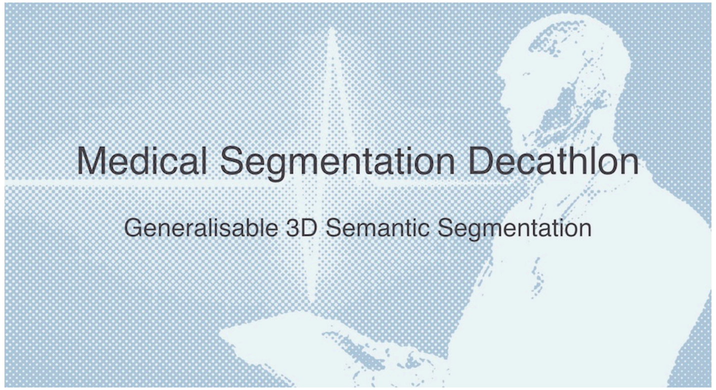


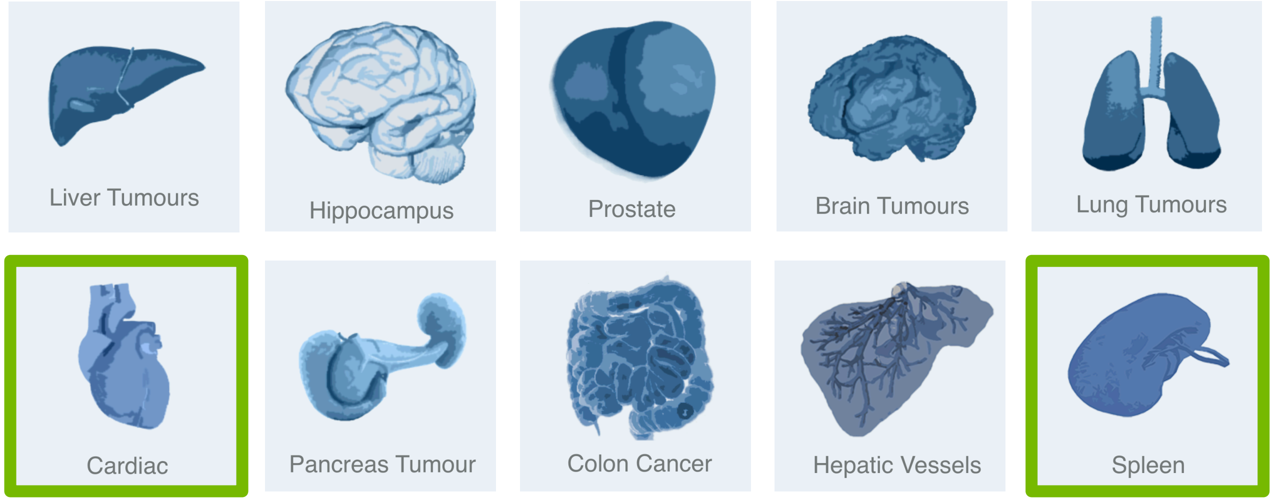
Çiçek, Ö., Abdulkadir, A., Lienkamp, S.S., Brox, T. and Ronneberger, O., 2016, October. 3D U-Net: learning dense volumetric segmentation from sparse annotation. In International conference on medical image computing and computer-assisted intervention (pp. 424-432). Springer, Cham.

DATASETS

Medical Segmentation Decathlon (MSD)

- 10 different applications of 3D medical image segmentation





BASELINE

Configuration

- Model
  - 3D U-Net in PyTorch 1.6.0
- Training configuration on 32GB Tesla V100 GPU
  - Patch based training (e.g. 128^3, or 160^3)
  - Batch size 2, 4, or 8 etc.
  - Adam optimizer with constant learning rate
- Metric $\text{Dice's Score} = \dfrac{2 \cdot |Y \cap \tilde{Y}|}{|Y| + |\tilde{Y}|}$
- Loss $\text{Dice's Loss} = 1.0 - \dfrac{2 \cdot \sum_i y_i \hat{y}_i}{\sum_i {y_i}^2 + \sum_i {\hat{y}_i}^2}$

Ground Truth $y_i \in Y$

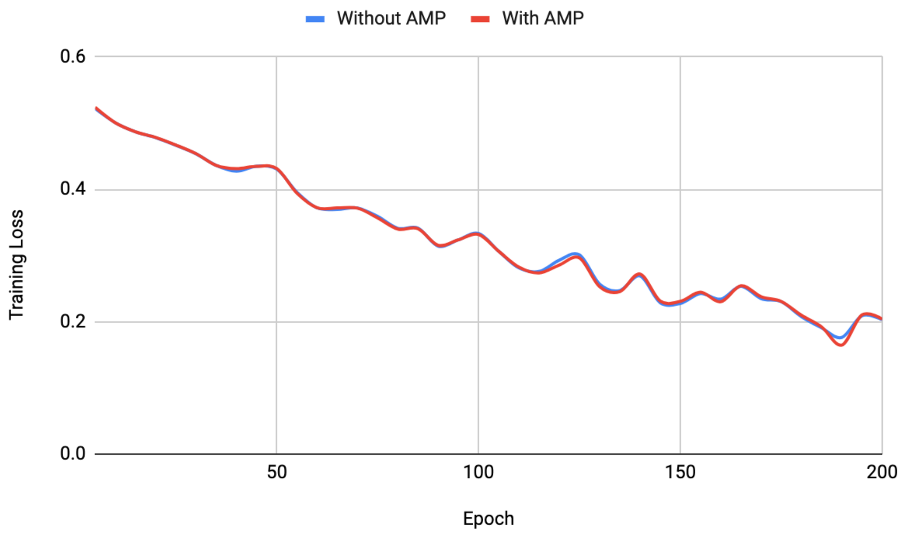
Model Prediction $\tilde{y}_i \in \tilde{Y}$

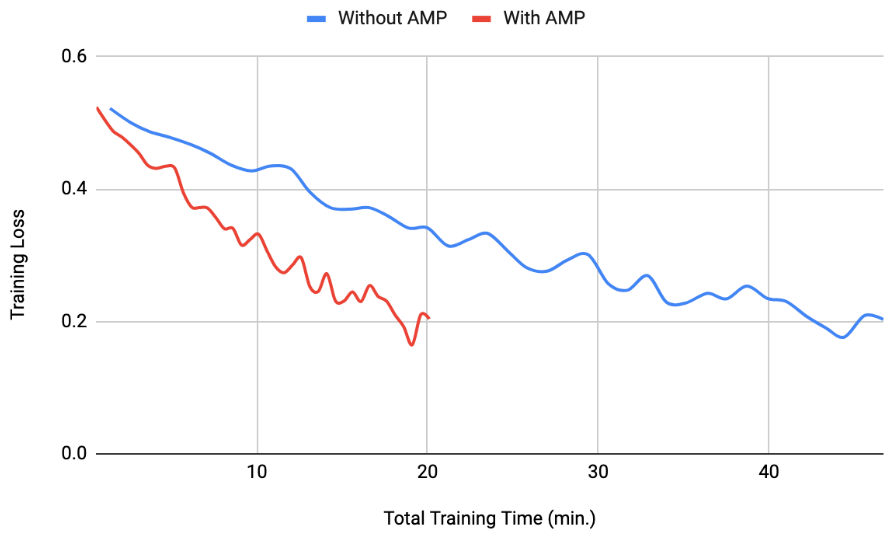
Probability after Softmax $\hat{y}_i \in \tilde{Y}$

TRAINING PERFORMANCE

Effectiveness

- Left atrium segmentation in 3D MRI (using fixed random seeds)



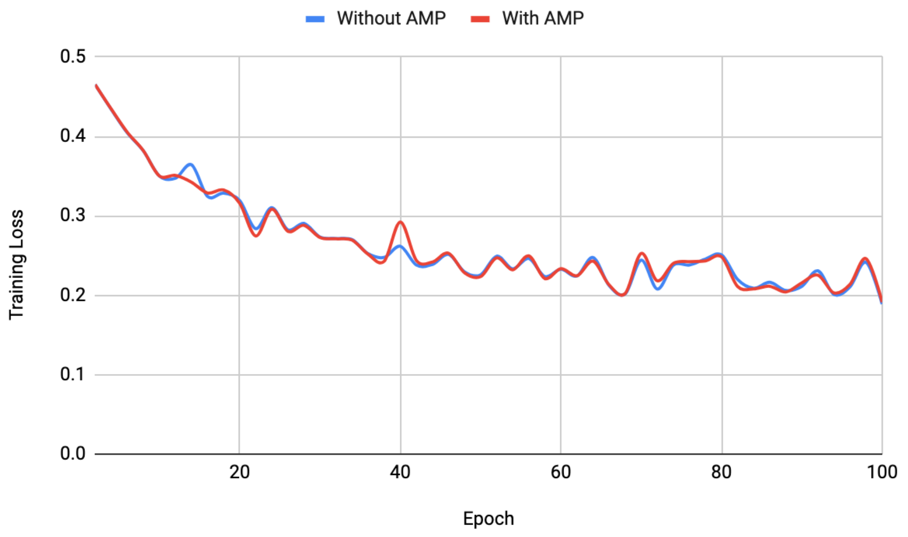


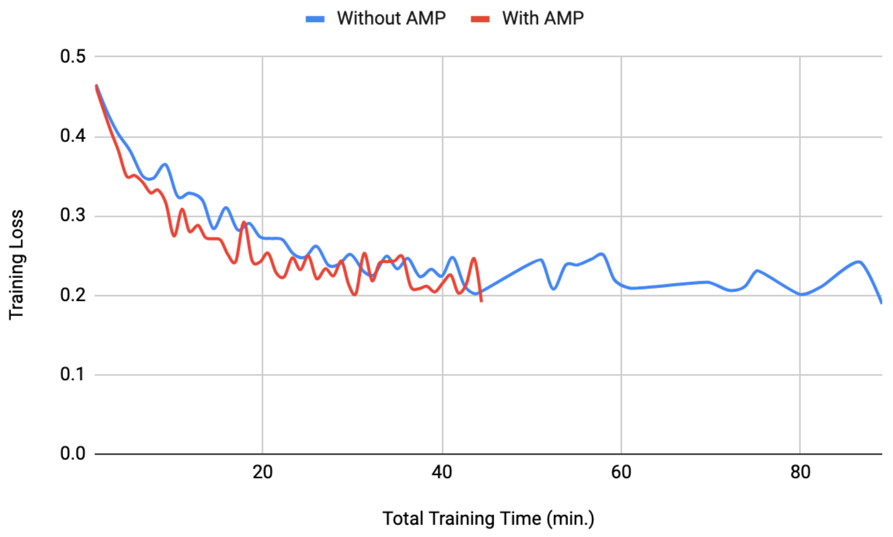
- Automatic Mixed Precision (AMP)

TRAINING PERFORMANCE

Effectiveness

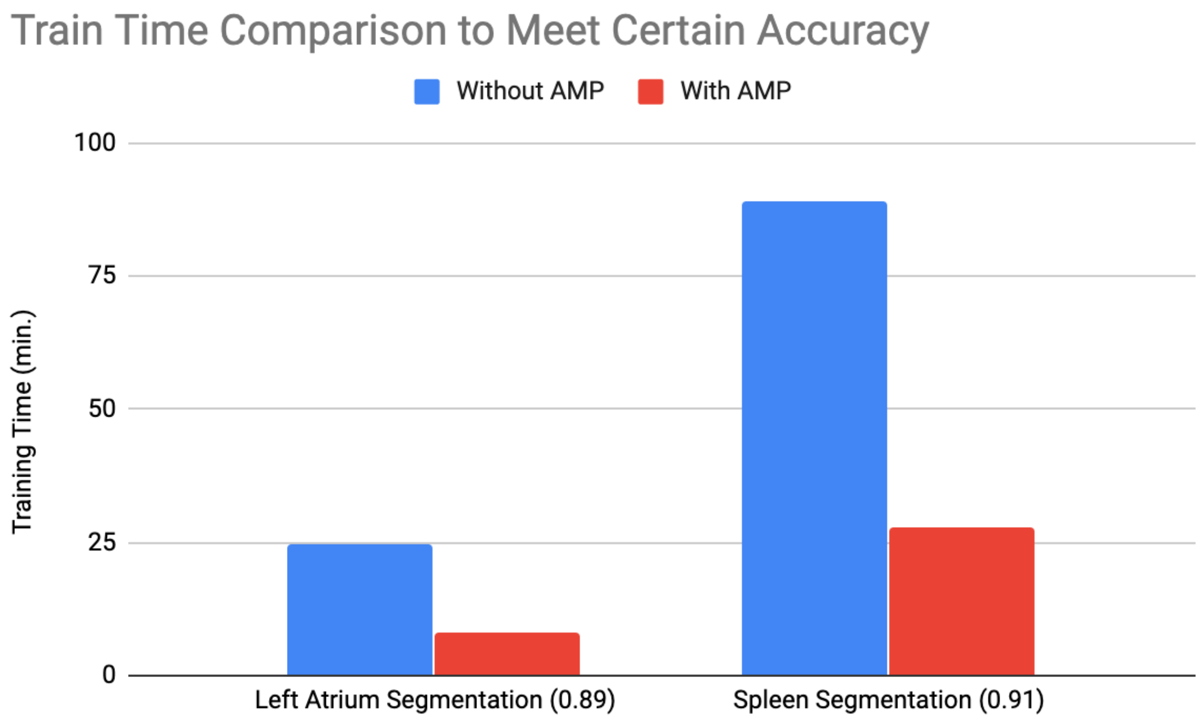
- Spleen segmentation in 3D CT (using fixed random seeds)





TRAINING PERFORMANCE

Accuracy



SUMMARY

High-level Network Breakdown

Method	Total	Forward	Backward	Loss	Optimizer
Training Patch Size 128 x 128 x 128 | **Batch Size 2**					
Non-AMP	0.362 s	0.081 s	0.252 s	0.001 s	0.028 s
AMP	0.181 s	0.062 s	0.091 s	0.001 s	0.027 s
Speed-up	x 2.0	x 1.3	x 2.8	x 1.0	x 1.0
Training Patch Size 128 x 128 x 128 | **Batch Size 8**					
Non-AMP	1.381 s	0.340 s	1.009 s	0.002 s	0.030 s
AMP	0.515 s	0.150 s	0.337 s	0.002 s	0.026 s
Speed-up	x 2.7	x 2.3	x 3.0	x 1.0	x 1.2

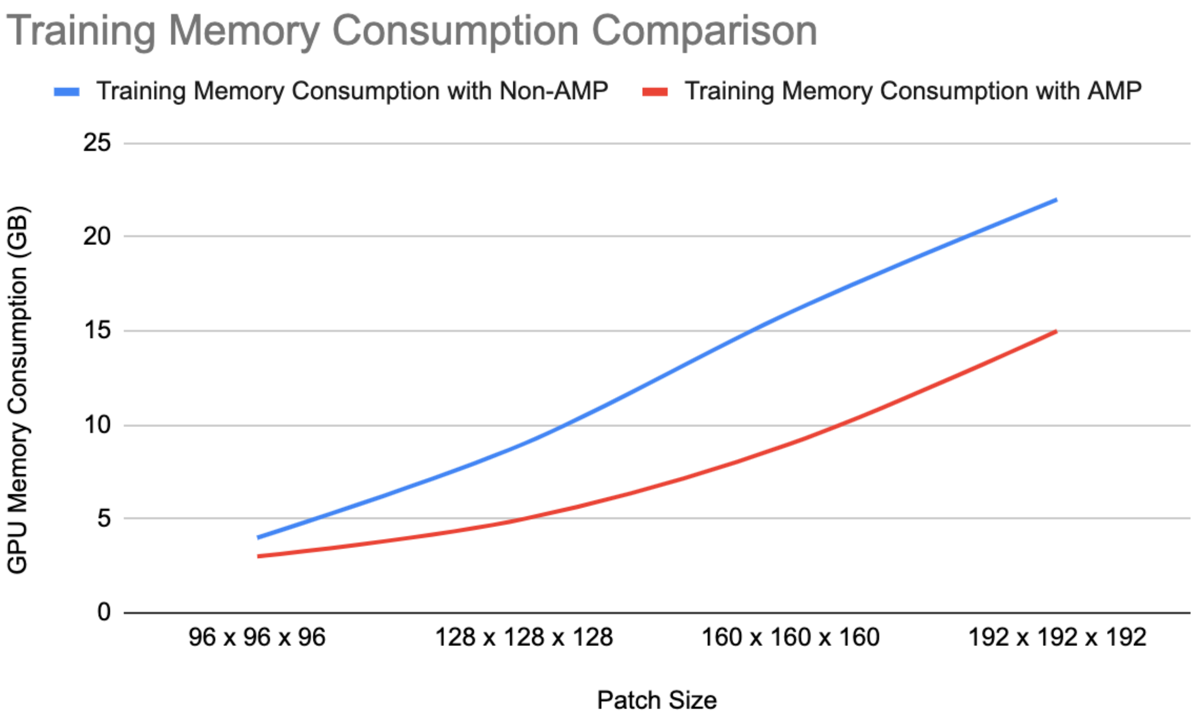
SUMMARY

High-level Network Breakdown

Method	Total	Forward	Backward	Loss	Optimizer
Training Patch Size 128 x 128 x 128 | Batch Size 8					
Non-AMP	1.381 s	0.340 s	1.009 s	0.002 s	0.030 s
AMP	0.515 s	0.150 s	0.337 s	0.002 s	0.026 s
Speed-up	x 2.7	x 2.3	x 3.0	x 1.0	x 1.2
Training Patch Size 160 x 160 x 160 | Batch Size 8					
Non-AMP	1.950 s	0.683 s	1.238 s	0.004 s	0.025 s
AMP	1.010 s	0.294 s	0.685 s	0.004 s	0.027 s
Speed-up	x 1.9	x 2.3	x 1.8	x 1.0	x 0.9

TRAINING PERFORMANCE

Efficiency



CONCLUSIONS

Mixed precision training is useful for 3D network training.

- Drop-in replacement when utilizing native PyTorch AMP library
  - Only need to change few lines of code
- For the same model and batch size, we can get more than 2.0x speed up
- With an optimized data flow, utilizing mixed precision training allows us training a bigger 3D model, which is the key toward better accuracy
- AMP helps usually when model capacity is large, or batch size is large

NVIDIA

Thank you!