Neural Architecture Search in Large-Scale 3D Medical Image Analysis

Dong Yang
NVIDIA

WACV 2020

Medical Image Analysis

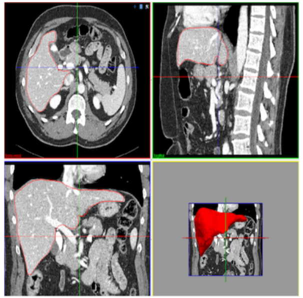
- What?
    - To gain high-level understanding from medical images
- Why?
    - Disease diagnosis, treatment planning and surgery guidance
- How?

Medical Image Analysis

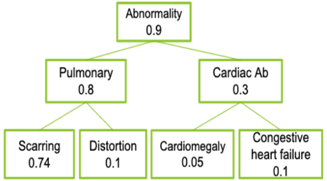
CT | MRI | X-Ray | US | Microscopy | Pathology | PET | OCT | EHR

Clinical Expertise + Artificial Intelligence / Deep Learning

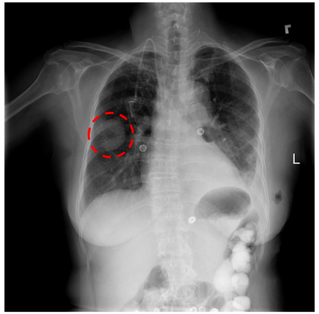
Segmentation



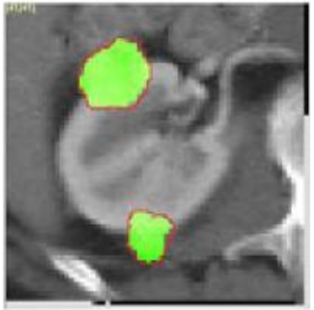
Classification



Detection



Survival Model

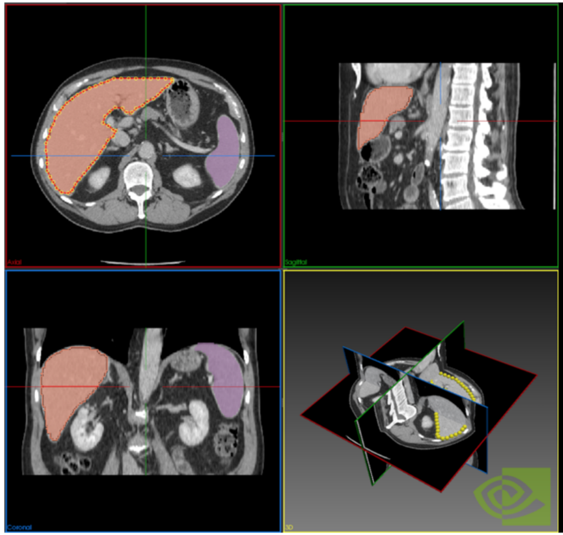


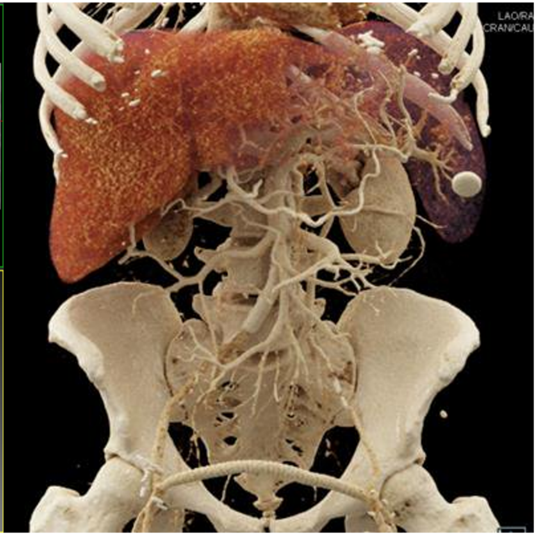
NLP

Findings:
There are diffuse bilateral interstitial and alveolar opacities consistent with chronic obstructive lung disease and bullous emphysema.
There are irregular opacities in the left lung apex, that could represent a cavitary lesion in the left lung apex.
There are streaky opacities in the right upper lobe, XXXX scarring.
...
Impression:
1. Bullous emphysema and interstitial fibrosis.
2. Probably scarring in the left apex, although difficult to exclude a cavitary lesion
...

Case Study - 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

Background

- U-Shape Network
  - One of the most popular and effective architecture styles in medical imaging. Since U-Net was proposed in 2015, various U-shape networks are proposed and achieve excellent performance.
  - Recently, nnUNet (variant of U-Net) won the *Medical Segmentation Decathlon* (MSD'18) and *Kidney and Tumor Segmentation Challenge* (KiTS'19).
- Neural Architecture Search (NAS)
  - Design network automatically instead of manually.
  - Architectures from NAS have shown superior performance (i.e. accuracy, latency, or model size) compared with manually designed ones.
- Objective: to achieve optimal architectures for 3D medical image segmentation

Ronneberger, et al. U-Net: Convolutional Networks for Biomedical Image Segmentation, MICCAI'15

Background

- Current NAS methods can be divided into the following categories
    - Reinforcement learning (RL) based search (NASNet)
    - Evolutionary algorithm (EA) based search (AmoebaNet)
    - Gradient based search (DARTS)
    - One-shot NAS (SMASH)
- However, literatures mainly focus on 2D image classification

Zoph, et al. Learning transferable architectures for scalable image recognition, CVPR'18.
Real, et al. Regularized evolution for image classifier architecture search, AAAI'19.
Liu, et al. Darts: Differentiable architecture search, ICLR'18.
Brock, et al. SMASH: one-shot model architecture search through hypernetworks, ICLR'18.

Contents

- Gradient-based searching
  - “V-NAS” (3DV’19)
- Hybrid searching
  - “C2FNAS” (CVPR’20)

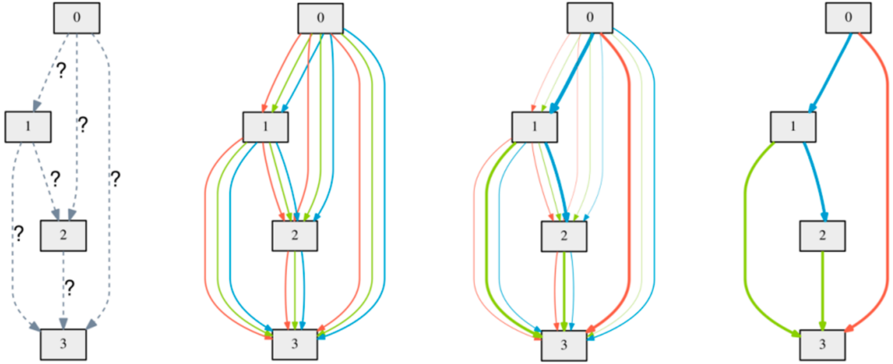
Contents

- Gradient-based searching
    - “V-NAS” (3DV’19)
- Hybrid searching
    - “C2FNAS” (CVPR’20)

V-NAS

Operation Search

- “DARTS”



(a) (b) (c) (d)

Liu, et al. Darts: Differentiable architecture search, ICLR’18.

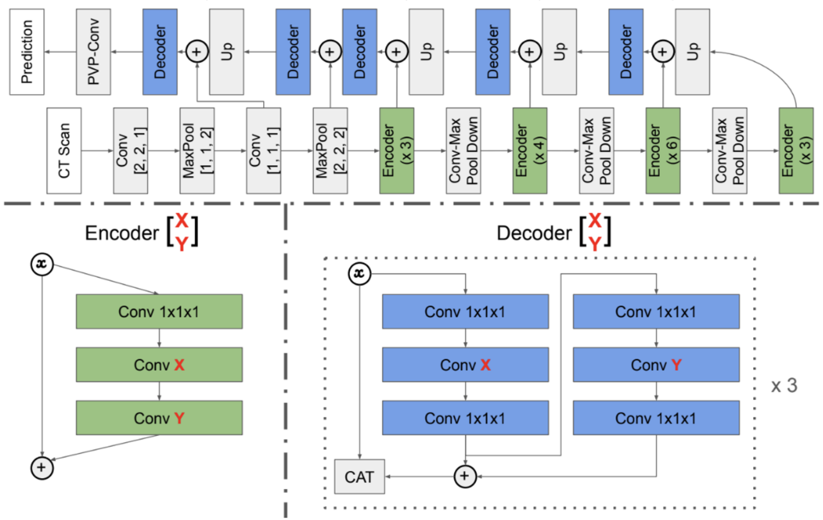
V-NAS

Operation Search

- Differentiable Neural Architecture Search
  - Started from 3D U-Net
  - Search for optimal convolution operations
    - 3D convolutions
    - 2D convolutions
    - Pseudo-3D convolutions (2D plus 1D)

V-NAS

Operation Search

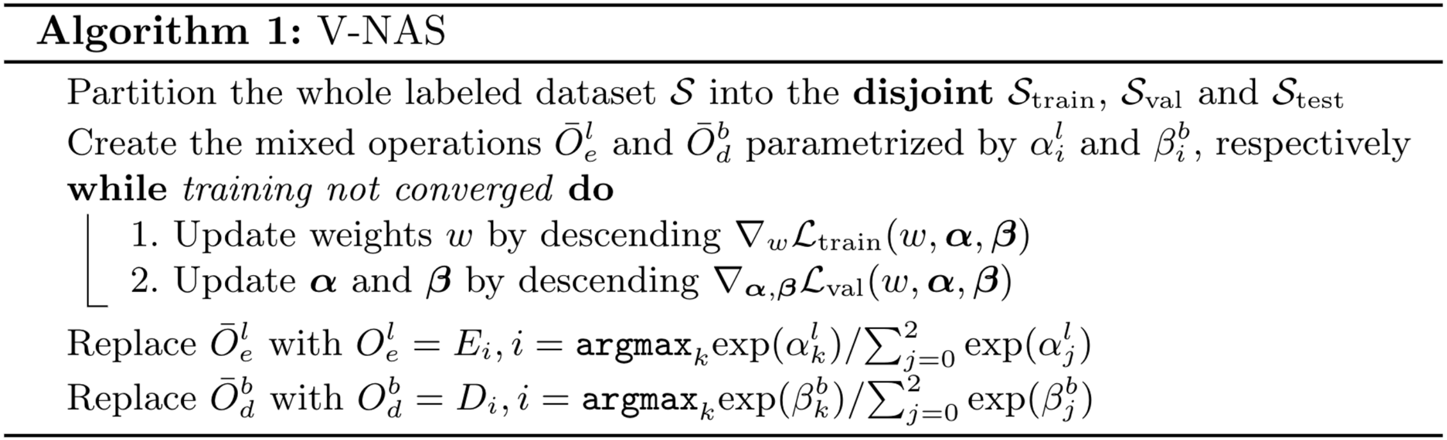


Encoder Search Space The set of possible Encoder architecture is denoted as $\mathcal{E}$, which includes the following 3 choices (*c.f.*, Fig.1 for Encoder $\begin{bmatrix}\text{X}\\\text{Y}\end{bmatrix}$):

$$\mathcal{E} = \{\underbrace{\text{Encoder}\begin{bmatrix}3\times3\times1\\1\times1\times1\end{bmatrix}}_{E_0\text{: 2D}}, \underbrace{\text{Encoder}\begin{bmatrix}3\times3\times3\\1\times1\times1\end{bmatrix}}_{E_1\text{: 3D}}, \underbrace{\text{Encoder}\begin{bmatrix}3\times3\times1\\1\times1\times3\end{bmatrix}}_{E_2\text{: P3D}}\} \qquad (1)$$

V-NAS

Operation Search



Algorithm 1: V-NAS

```
Partition the whole labeled dataset $\mathcal{S}$ into the **disjoint** $\mathcal{S}_{\text{train}}$, $\mathcal{S}_{\text{val}}$ and $\mathcal{S}_{\text{test}}$
Create the mixed operations $\bar{O}_e^l$ and $\bar{O}_d^b$ parametrized by $\alpha_i^l$ and $\beta_i^b$, respectively
while training not converged do
    1. Update weights $w$ by descending $\nabla_w \mathcal{L}_{\text{train}}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$
    2. Update $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ by descending $\nabla_{\boldsymbol{\alpha},\boldsymbol{\beta}} \mathcal{L}_{\text{val}}(w, \boldsymbol{\alpha}, \boldsymbol{\beta})$
Replace $\bar{O}_e^l$ with $O_e^l = E_i, i = \texttt{argmax}_k \exp(\alpha_k^l)/\sum_{j=0}^{2} \exp(\alpha_j^l)$
Replace $\bar{O}_d^b$ with $O_d^b = D_i, i = \texttt{argmax}_k \exp(\beta_k^b)/\sum_{j=0}^{2} \exp(\beta_j^b)$
```

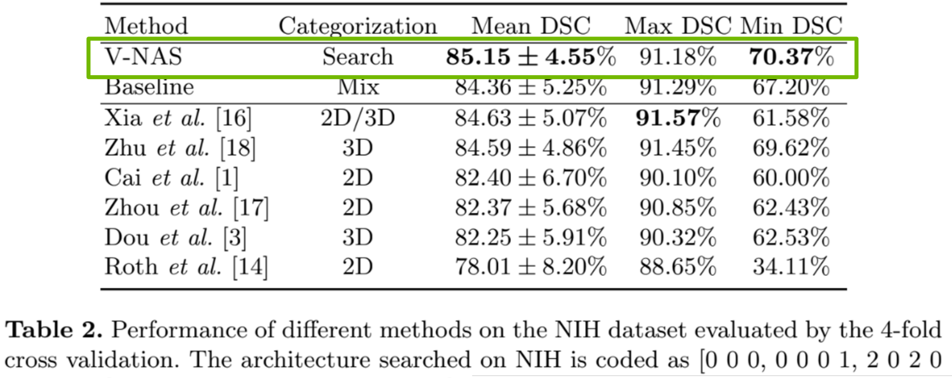
V-NAS

Operation Search

- Differentiable Neural Architecture Search
    - Searching is conducted during model training
    - Weights for different operations are updated on-the-fly
    - The searched network is finalized after training is converged
    - Re-training final network from scratch for optimal performance

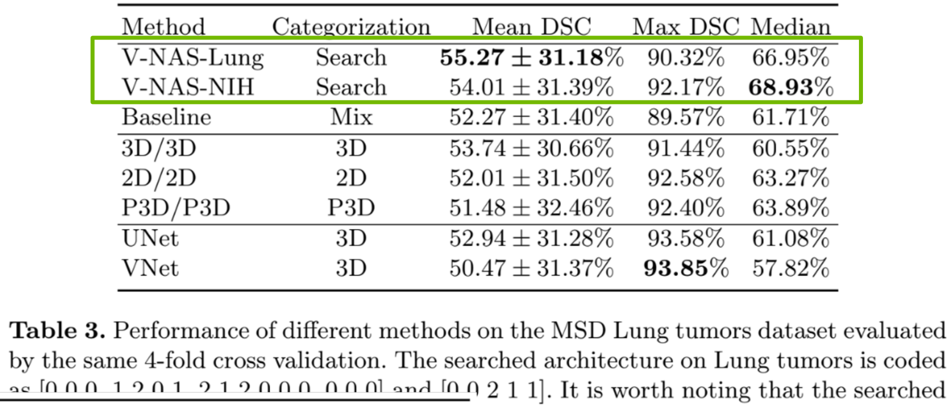
V-NAS

Experiments



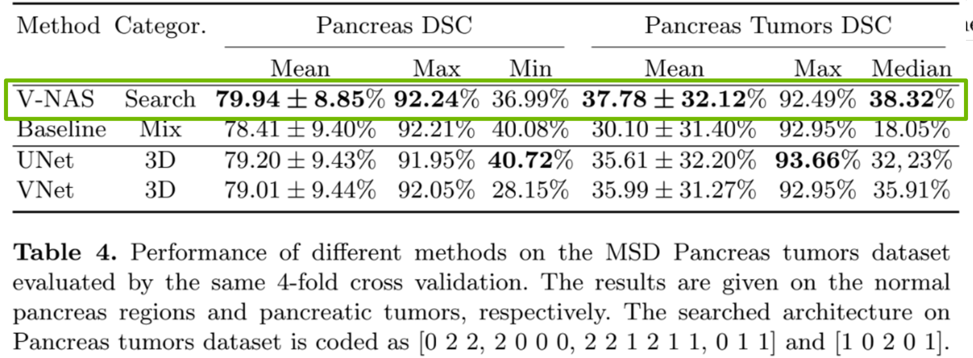
Method	Categorization	Mean DSC	Max DSC	Min DSC
V-NAS	Search	**85.15 ± 4.55%**	91.18%	**70.37%**
Baseline	Mix	84.36 ± 5.25%	91.29%	67.20%
Xia *et al.* [16]	2D/3D	84.63 ± 5.07%	**91.57%**	61.58%
Zhu *et al.* [18]	3D	84.59 ± 4.86%	91.45%	69.62%
Cai *et al.* [1]	2D	82.40 ± 6.70%	90.10%	60.00%
Zhou *et al.* [17]	2D	82.37 ± 5.68%	90.85%	62.43%
Dou *et al.* [3]	3D	82.25 ± 5.91%	90.32%	62.53%
Roth *et al.* [14]	2D	78.01 ± 8.20%	88.65%	34.11%

Table 2. Performance of different methods on the NIH dataset evaluated by the 4-fold cross validation. The architecture searched on NIH is coded as [0 0 0, 0 0 0 1, 2 0 2 0 2 2, 0 0 0] for the 16 Encoder cells, and [0



Method	Categorization	Mean DSC	Max DSC	Median
V-NAS-Lung	Search	**55.27 ± 31.18%**	90.32%	66.95%
V-NAS-NIH	Search	54.01 ± 31.39%	92.17%	**68.93%**
Baseline	Mix	52.27 ± 31.40%	89.57%	61.71%
3D/3D	3D	53.74 ± 30.66%	91.44%	60.55%
2D/2D	2D	52.01 ± 31.50%	92.58%	63.27%
P3D/P3D	P3D	51.48 ± 32.46%	92.40%	63.89%
UNet	3D	52.94 ± 31.28%	93.58%	61.08%
VNet	3D	50.47 ± 31.37%	**93.85%**	57.82%

Table 3. Performance of different methods on the MSD Lung tumors dataset evaluated by the same 4-fold cross validation. The searched architecture on Lung tumors is coded as [0 0 0, 1 2 0 1, 2 1 2 0 0 0, 0 0 0] and [0 0 2 1 1]. It is worth noting that the searched eralized to the Lung tumors dataset.

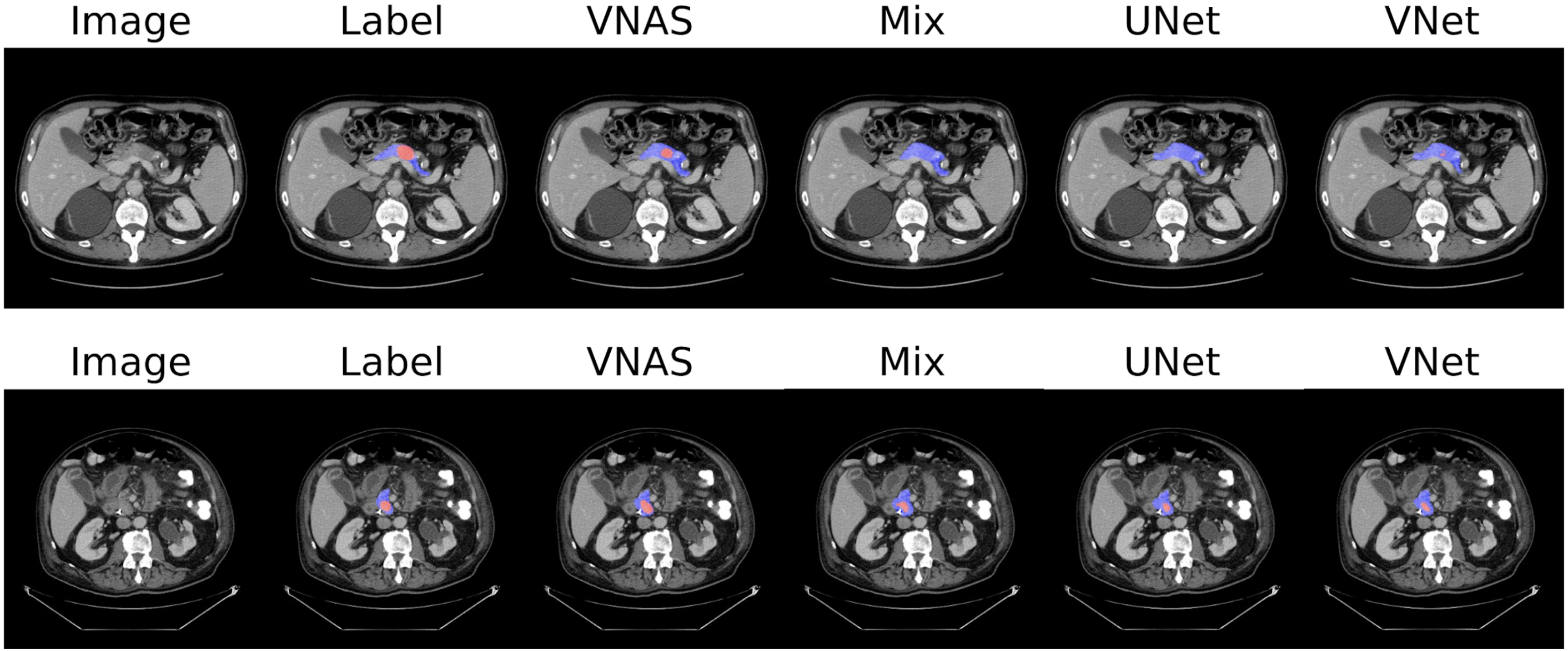


Method	Categor.	Pancreas DSC			Pancreas Tumors DSC		
		Mean	Max	Min	Mean	Max	Median
V-NAS	Search	**79.94 ± 8.85%**	**92.24%**	36.99%	**37.78 ± 32.12%**	92.49%	**38.32%**
Baseline	Mix	78.41 ± 9.40%	92.21%	40.08%	30.10 ± 31.40%	92.95%	18.05%
UNet	3D	79.20 ± 9.43%	91.95%	**40.72%**	35.61 ± 32.20%	**93.66%**	32, 23%
VNet	3D	79.01 ± 9.44%	92.05%	28.15%	35.99 ± 31.27%	92.95%	35.91%

Table 4. Performance of different methods on the MSD Pancreas tumors dataset evaluated by the same 4-fold cross validation. The results are given on the normal pancreas regions and pancreatic tumors, respectively. The searched architecture on Pancreas tumors dataset is coded as [0 2 2, 2 0 0 0, 2 2 1 2 1 1, 0 1 1] and [1 0 2 0 1].

V-NAS

Results



Contents

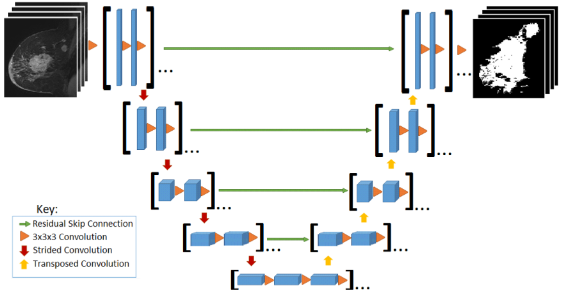
- Gradient-based searching
  - “V-NAS” (3DV’19)
- Hybrid searching
  - “C2FNAS” (CVPR’20)

Our Proposed Approach

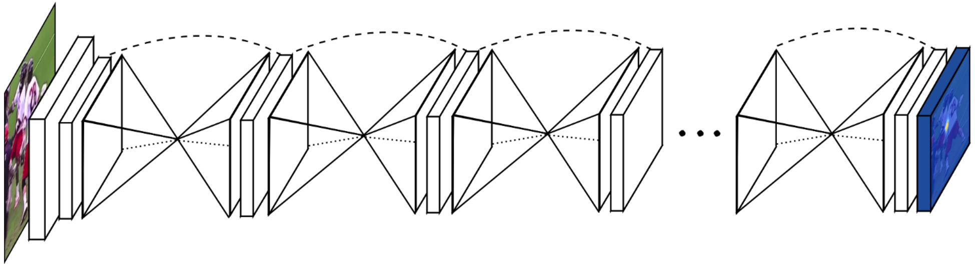
- ‘’C2FNAS”
    - Coarse-to-fine neural architecture search
- Multi-Level Searching Strategy
    - Step 1 – *Macro-level*
        - Evolutionary algorithm for macro-level
    - Step 2 – *Micro-level*
        - Super-Net training for micro-level
    - Step 3 – *Compound Scaling*

Search Space – Step 1/3

- Search at Macro-Level (**Network**)
  - Network Shape: According to the order of down-sample layers and up-sample layers, we can divide networks into U-Net-kind and Stacked-Hourglass-kind
  - Layer Assignment: Different from a symmetric U-Net design, we try to search for different assignments of layers, which make it asymmetric



U-Net



Stacked-Hourglass

Newell, et al. Stacked hourglass networks for human pose estimation, ECCV'16.

Search Space – Step 2/3

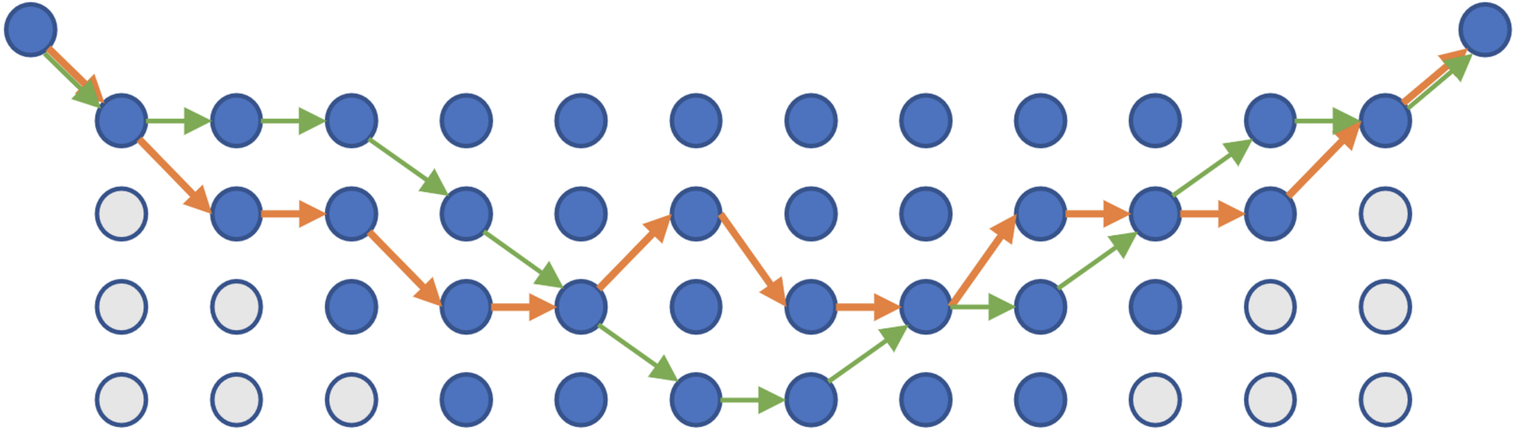
- Search at Micro-Level (**Cell**)
  - Next, search for a replacement for *op* in each cell, each *op* can be selected from
    - 3x3x3 3D Conv.
    - 5x5x5 3D Conv.
    - 3x3x1 Pseudo 3D Conv.
    - 5x5x1 Pseudo 3D Conv.
    - 3x3x3 3D Conv. with Dilation = 2
    - 5x5x5 3D Conv. with Dilation = 2

Search Space – Step 3/3

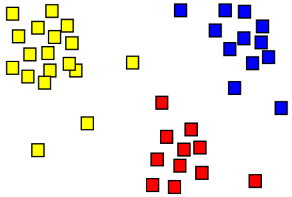
- Compound Scaling
    - To better balance the performance and model size, we scale the **patch size**, **cell numbers**, and **filter numbers**, inspired by EfficientNet (STOA performance on ImageNet)

Tan, et al. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, ICML'19.

Search Space - Network Level

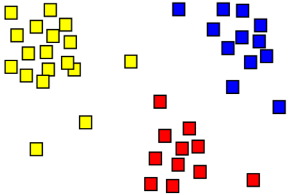


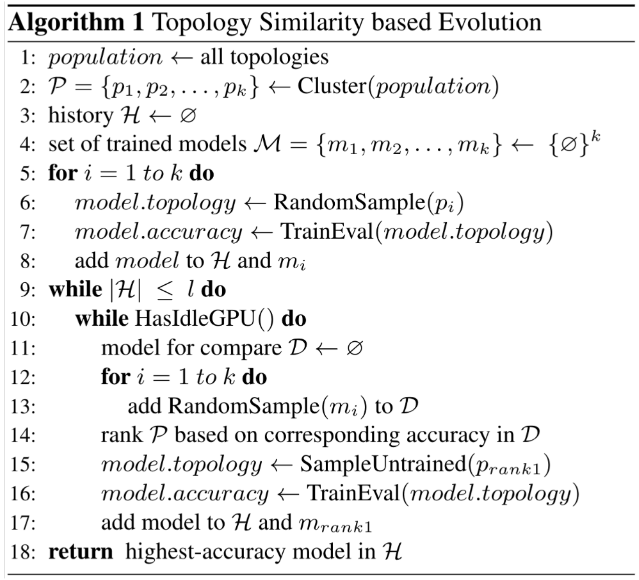
Search Method



- **Network-level**: Small search space (at most one thousand candidates)
  - Intuitively, models with similar architectures should have similar performances
  - We propose a clustering-based evolutionary algorithm
    - Step 1 - Search space is divided into K clusters, based on their similarity on network architectures.
    - Step 2 - Each cluster can generate child net based on a probability, we train those nets and update performance history for each cluster.
    - Step 3 - A net is random sampled from each cluster. By comparing these nets performance, we re-rank the clusters and assigned corresponding probability.

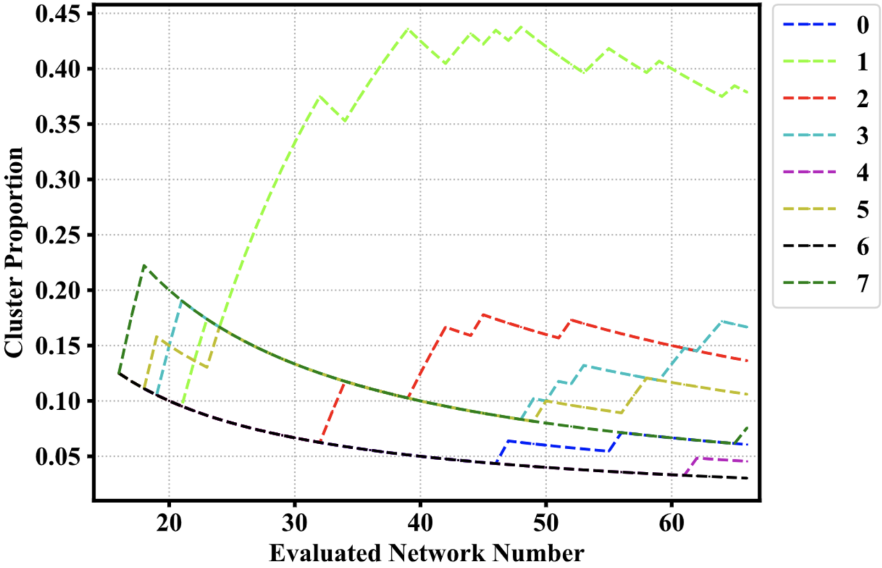
Search Method





Algorithm 1 Topology Similarity based Evolution

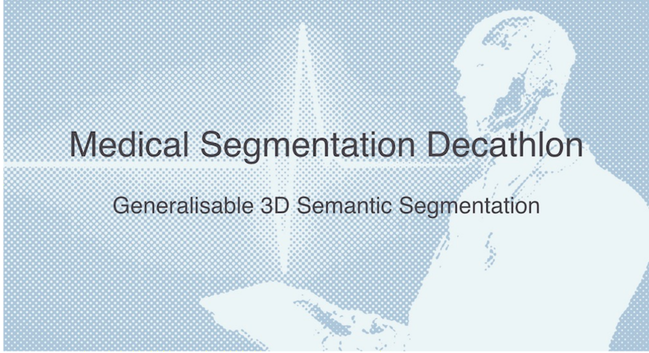
```
population ← all topologies
$\mathcal{P} = \{p_1, p_2, \dots, p_k\} \leftarrow \text{Cluster}(population)$
history $\mathcal{H} \leftarrow \varnothing$
set of trained models $\mathcal{M} = \{m_1, m_2, \dots, m_k\} \leftarrow \{\varnothing\}^k$
for $i = 1$ to $k$ do
    $model.topology \leftarrow \text{RandomSample}(p_i)$
    $model.accuracy \leftarrow \text{TrainEval}(model.topology)$
    add $model$ to $\mathcal{H}$ and $m_i$
while $|\mathcal{H}| \le l$ do
    while HasIdleGPU() do
        model for compare $\mathcal{D} \leftarrow \varnothing$
        for $i = 1$ to $k$ do
            add RandomSample($m_i$) to $\mathcal{D}$
        rank $\mathcal{P}$ based on corresponding accuracy in $\mathcal{D}$
        $model.topology \leftarrow \text{SampleUntrained}(p_{rank1})$
        $model.accuracy \leftarrow \text{TrainEval}(model.topology)$
        add model to $\mathcal{H}$ and $m_{rank1}$
return highest-accuracy model in $\mathcal{H}$
```



Search Method

- **Cell-level:** Search space can be as large as more than millions of candidates, thus EA/RL based method can be less effective.
    - Thus, we treat each candidate as a sub-graph of a super-net.
    - We train the super-net by sampling paths uniformly and use it to predict the performance for each candidate. (one-shot NAS)
- **Compound scale:** Small search space, based on EfficientNet, grid search can be an alternative solution.

Datasets

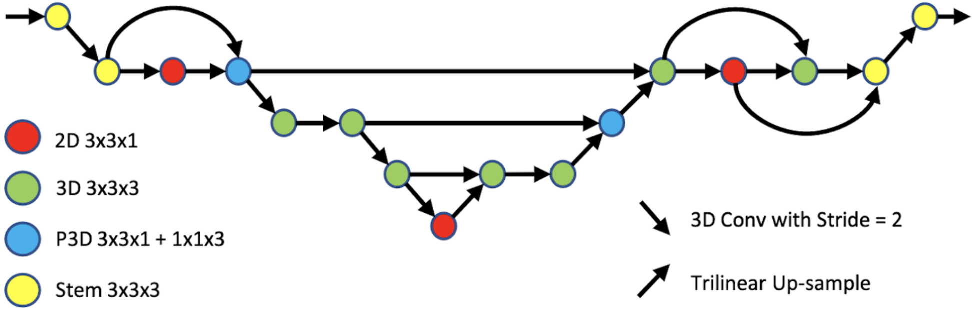

Medical Segmentation Decathlon
Generalisable 3D Semantic Segmentation

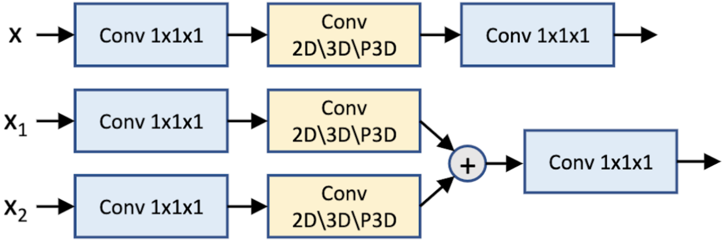
- Medical Segmentation Decalthlon (MSD)

Task	Training	Test
01 - Brain Tumor Segmentation	484	266
06 - Lung Tumor Segmentation	63	32
07 - Pancreas and Tumor Segmentation	281	139

- We only have access to training labels, and testing results can only be obtained through submitting to test server, once a day at most, which guarantees that our model is not "over-fitting" the test data

Final Architecture (Searched on Pancreas Seg.)





Model Training Details

- Pre-processing
  - Intensity clipping and standard normalization
- Augmentation
  - Random rotation and flipping
- Optimizer
  - SGD
- Loss
  - Soft dice loss and cross-entropy

Results on Test Set – Dice's Score (DSC)

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

Brain Tumor Segmentation	DSC Edema	DSC Non-enhancing Tumor	DSC Enhancing Tumor	Average
NV_DLMED	67.52	45.00	68.01	60.18
nnUNet	67.71	47.73	68.16	61.20
Ours	**68.74**	**48.22**	**69.19**	**62.05**

Lung Tumor Segmentation	DSC Tumor	**Pancreas and Tumor Segmentation**	DSC Pancreas	DSC Tumor	Average
NV_DLMED	52.15	NV_DLMED	78.42	38.48	58.45
nnUNet	69.20	nnUNet	79.53	52.27	65.90
Ours	**70.94**	Ours	**80.41**	**53.67**	**67.04**

nnUNet is the winner of Medical Segmentation Decathlon (MSD) last year, and NV_DLMED (previous entry) was second place.

Model Comparison

Model	3D U-Net	V-Net	AH-Net	nnU-Net	Ours
Params (M)	16.32	45.61	27.11	10.36	**3.91**
FLOPS (G)	802.9	322.5	**29.5**	202.25	184.8

It is noticeable that our model is much more **compact** compared with other models, and also **fewer FLOPS** compared with other 3D models. The evaluation is done with input size (1,4,96,96,96), and output for 4 classes. AH-Net has a much smaller FLOPS because it uses a 2D encoder.

Analysis

nnUNet (“state-of-the-art”) is a 2D U-Net, a 3D U-Net, or Cascaded 3D U-Nets, with well-tuned hyperparameters, it applies many tricks like many kinds of data augmentation, test-time augmentation, a complicated learning rate schedule, coarse-to-fine, model ensemble, and so on.

While our method mainly focus on improving the model itself with much simpler settings. nnUNet can be considered as an upper-bound of past U-Net design with many engineering tricks. Thus, our model truly beat the past U-Net design by beating nnUNet.

Search Cost Estimation (16 GB V100)

- **Train a Network**: 30 GPU hrs (3~4 hrs with 8GPU)
- **Net-level Search**: 50 networks are evaluted => 1500 GPU hrs
- **Cell-level Search**: SuperNet training 80 GPU hrs, search (evaluate) ~300 GPU hrs
- **Compound Scale**: ~20 Networks and thus 600 GPU hrs
- **In total:** 1500 + 80 + 300 + 600 = ~2480 GPU hrs = ~180 GPU days
- **In practice:** Using 32 GPUs in parallel, searching is done within 100 hours on average

- We are looking for some ways to reduce the search cost, like increasing GPU utilization, and reduce training iterations, and so on.

Case Study

	NV-DLMED	Ours
Pancreas (Red)	81.87%	**84.14%**
Tumor (Green)	0.00%	**74.77%**

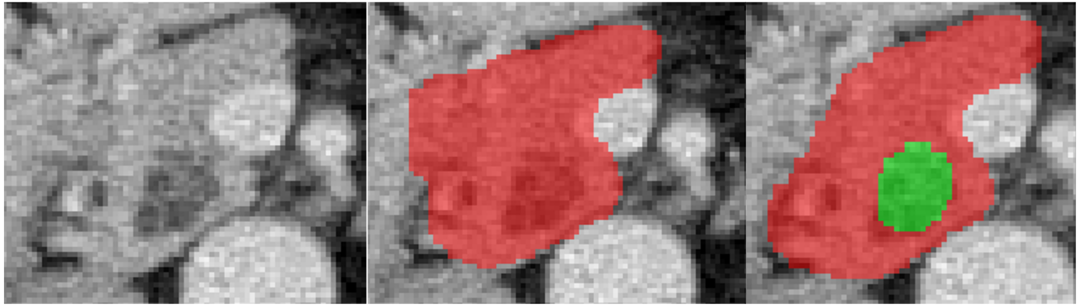


Image NV-DLMED Ours

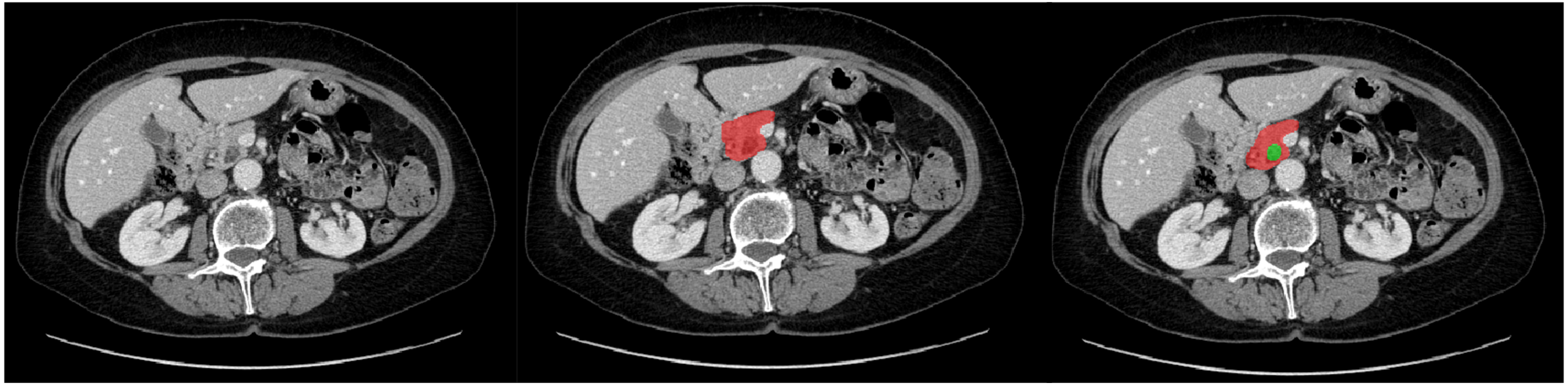


Image NV-DLMED Ours

Case Study

	NV-DLMED	Ours
Edema (Red)	70.58%	**78.61%**
Non-enhancing Tumor (Green)	10.18%	**51.51%**
Enhancing Tumor (Blue)	32.73%	**46.34%**

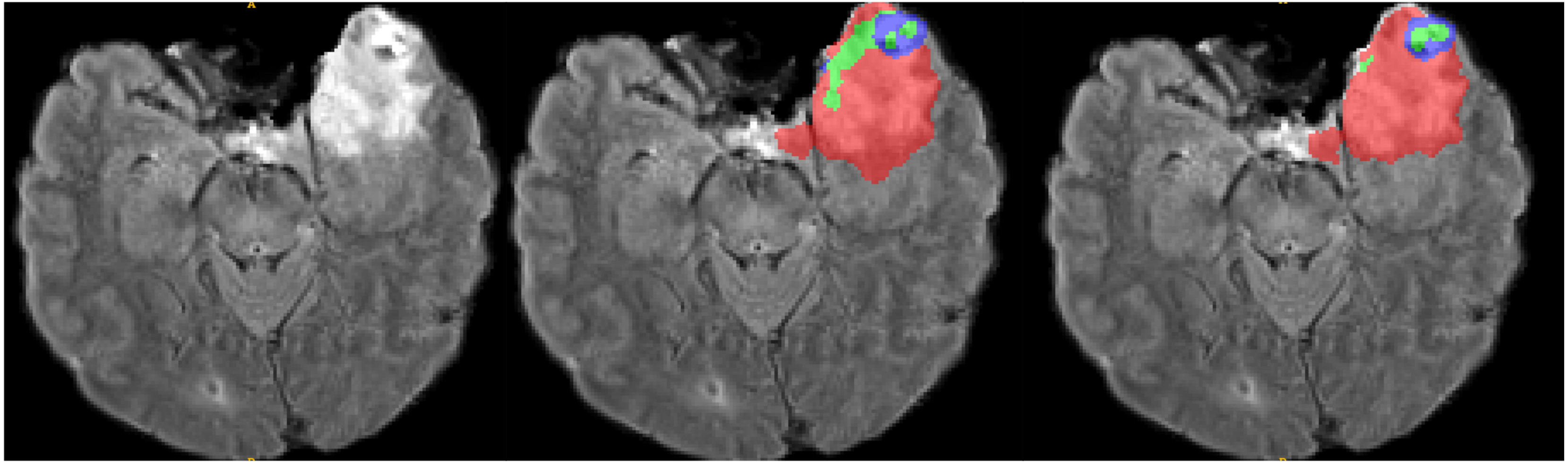


Image NV-DLMED Ours

Task	Brain			Heart	Liver		Pancreas		Prostate		Lung	Hippocampus		HepaticVessel		Spleen	Colon	Avg (Task)	Avg (Class)
Class	1	2	3	1	1	2	1	2	1	2	1	1	2	1	2	1	1		
CerebriuDIKU [20]	**69.52**	43.11	66.74	89.47	94.27	57.25	71.23	24.98	69.11	86.34	58.71	89.68	88.31	59.00	38.00	95.00	28.00	67.01	66.40
Lupin	66.15	41.63	64.15	91.86	94.79	61.40	75.99	21.24	72.73	87.62	54.61	89.66	88.26	60.00	47.00	94.00	9.00	65.61	65.89
NVDLMED [32]	67.52	45.00	68.01	92.46	95.06	71.40	78.42	38.48	69.36	86.66	52.15	87.97	86.71	63.00	64.00	96.00	56.00	72.73	71.66
K.A.V.athlon	66.63	46.62	67.46	91.72	94.74	61.65	74.97	43.20	73.42	87.80	60.56	89.83	88.52	62.00	63.00	**97.00**	36.00	71.51	70.89
nnU-Net [11]	67.71	47.73	68.16	**92.77**	**95.24**	**73.71**	79.53	52.27	**75.81**	**89.59**	69.20	**90.37**	**88.95**	63.00	69.00	96.00	56.00	76.39	75.00
C2FNAS-Panc	67.62	48.56	69.09	92.13	94.91	71.63	80.59	52.87	73.11	87.43	69.47	86.87	85.44	63.78	69.41	96.60	55.68	75.87	74.42
C2FNAS-Panc*	67.62	**48.60**	**69.72**	92.49	94.98	72.89	**80.76**	**54.41**	74.88	88.75	**70.44**	89.37	87.96	**64.30**	**71.00**	96.28	**58.90**	**76.97**	**75.49**

Table 1. Comparison with state-of-the-art methods on MSD challenge test set (number from MSD leaderboard). * denotes the 5-fold model ensemble. The numbers of tasks hepatic vessel, spleen, and colon from other teams are rounded. We also report the average on tasks and on targets respectively for an overall comparison across all tasks/targets.

Contributions

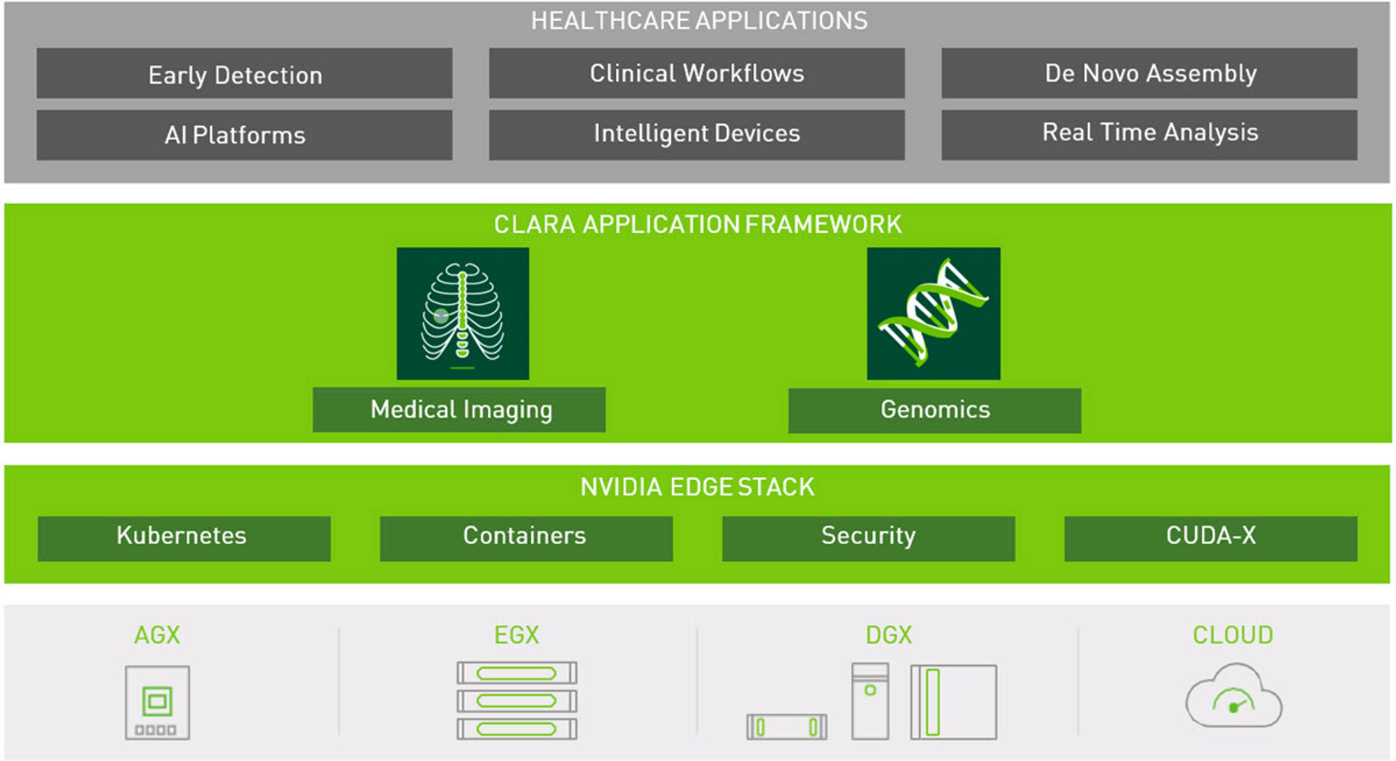
- Designed search spaces for 3D medical imaging segmentation, which leverages the merits of other established network design
- Designed different search methods for different search spaces, boosting searching efficiency
- Achieved architectures which beat previsous SOTA U-shape networks

References

- **[3DV’19]** Zhu, Z., Liu, C., Yang, D., Yuille, A. and Xu, D., 2019, September. **V-NAS: Neural Architecture Search for Volumetric Medical Image Segmentation.** In *2019 International Conference on 3D Vision (3DV)* (pp. 240-248). IEEE.

- **[CVPR’20]** Yu, Q., Yang, D., Roth, H., Bai, Y., Zhang, Y., Yuille, A.L. and Xu, D., 2020, June. **C2FNAS: Coarse-to-Fine Neural Architecture Search for 3D Medical Image Segmentation.** *In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR).*

NVIDIA Clara



HEALTHCARE APPLICATIONS

Early Detection

Clinical Workflows

De Novo Assembly

AI Platforms

Intelligent Devices

Real Time Analysis

CLARA APPLICATION FRAMEWORK

Medical Imaging

Genomics

NVIDIA EDGE STACK

Kubernetes

Containers

Security

CUDA-X

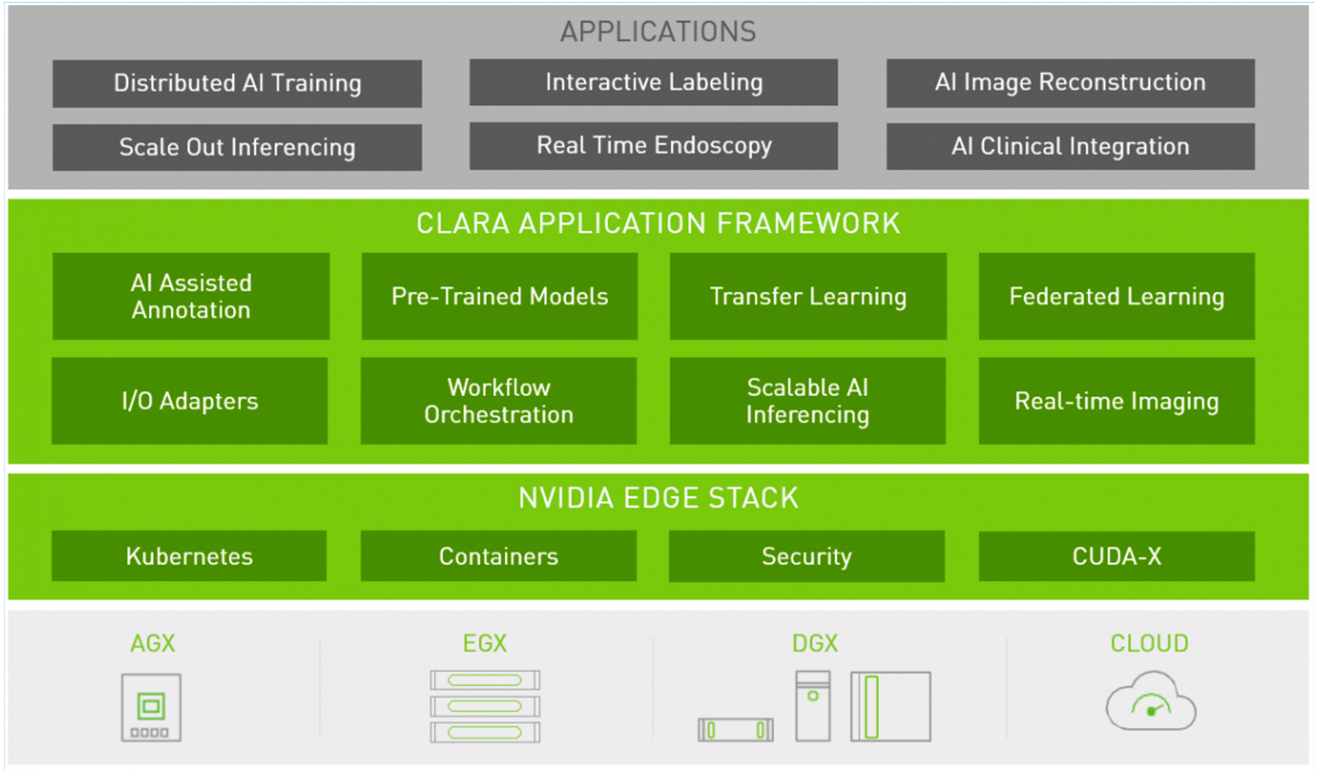
AGX

EGX

DGX

CLOUD

NVIDIA Clara Medical Imaging



APPLICATIONS

Distributed AI Training

Interactive Labeling

AI Image Reconstruction

Scale Out Inferencing

Real Time Endoscopy

AI Clinical Integration

CLARA APPLICATION FRAMEWORK

AI Assisted Annotation

Pre-Trained Models

Transfer Learning

Federated Learning

I/O Adapters

Workflow Orchestration

Scalable AI Inferencing

Real-time Imaging

NVIDIA EDGE STACK

Kubernetes

Containers

Security

CUDA-X

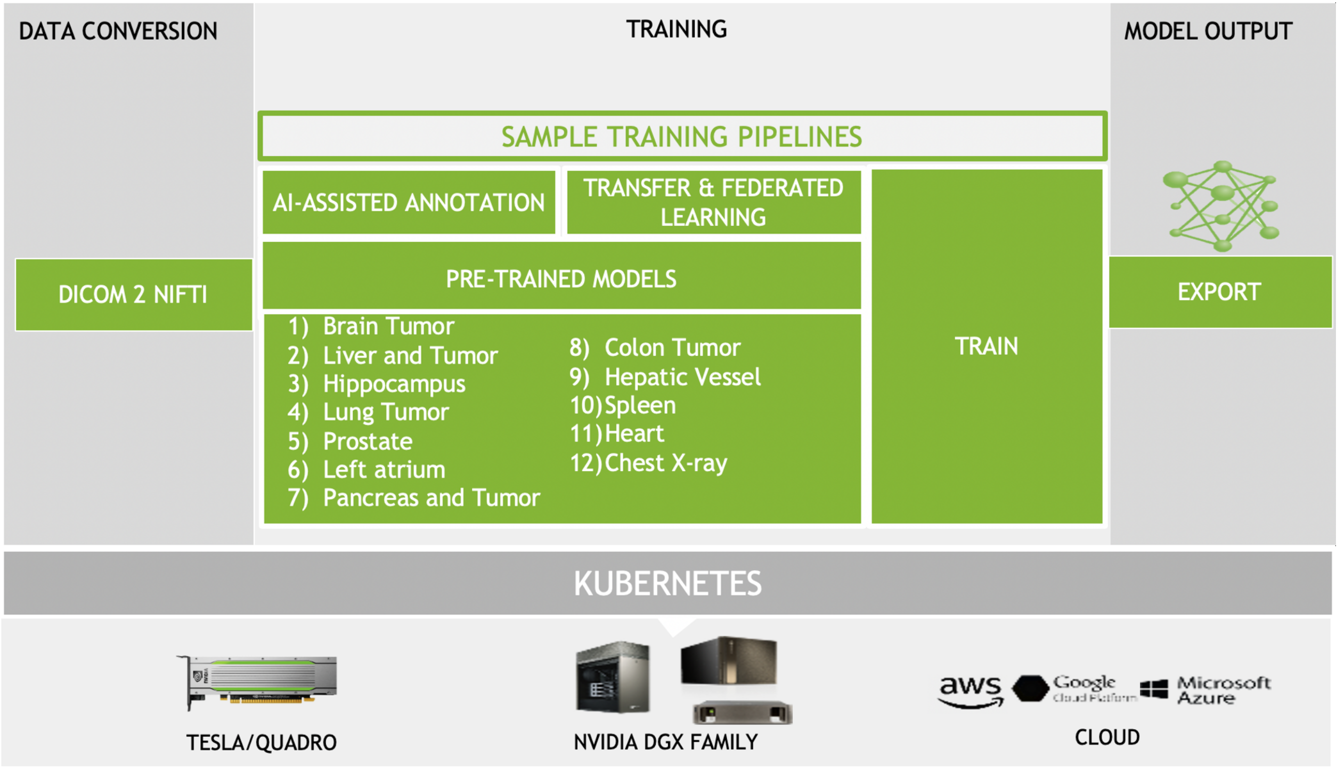
AGX

EGX

DGX

CLOUD

NVIDIA Clara Medical Imaging



DATA CONVERSION

DICOM 2 NIFTI

TRAINING

SAMPLE TRAINING PIPELINES

AI-ASSISTED ANNOTATION

TRANSFER & FEDERATED LEARNING

PRE-TRAINED MODELS

1) Brain Tumor
2) Liver and Tumor
3) Hippocampus
4) Lung Tumor
5) Prostate
6) Left atrium
7) Pancreas and Tumor
8) Colon Tumor
9) Hepatic Vessel
10) Spleen
11) Heart
12) Chest X-ray

TRAIN

MODEL OUTPUT

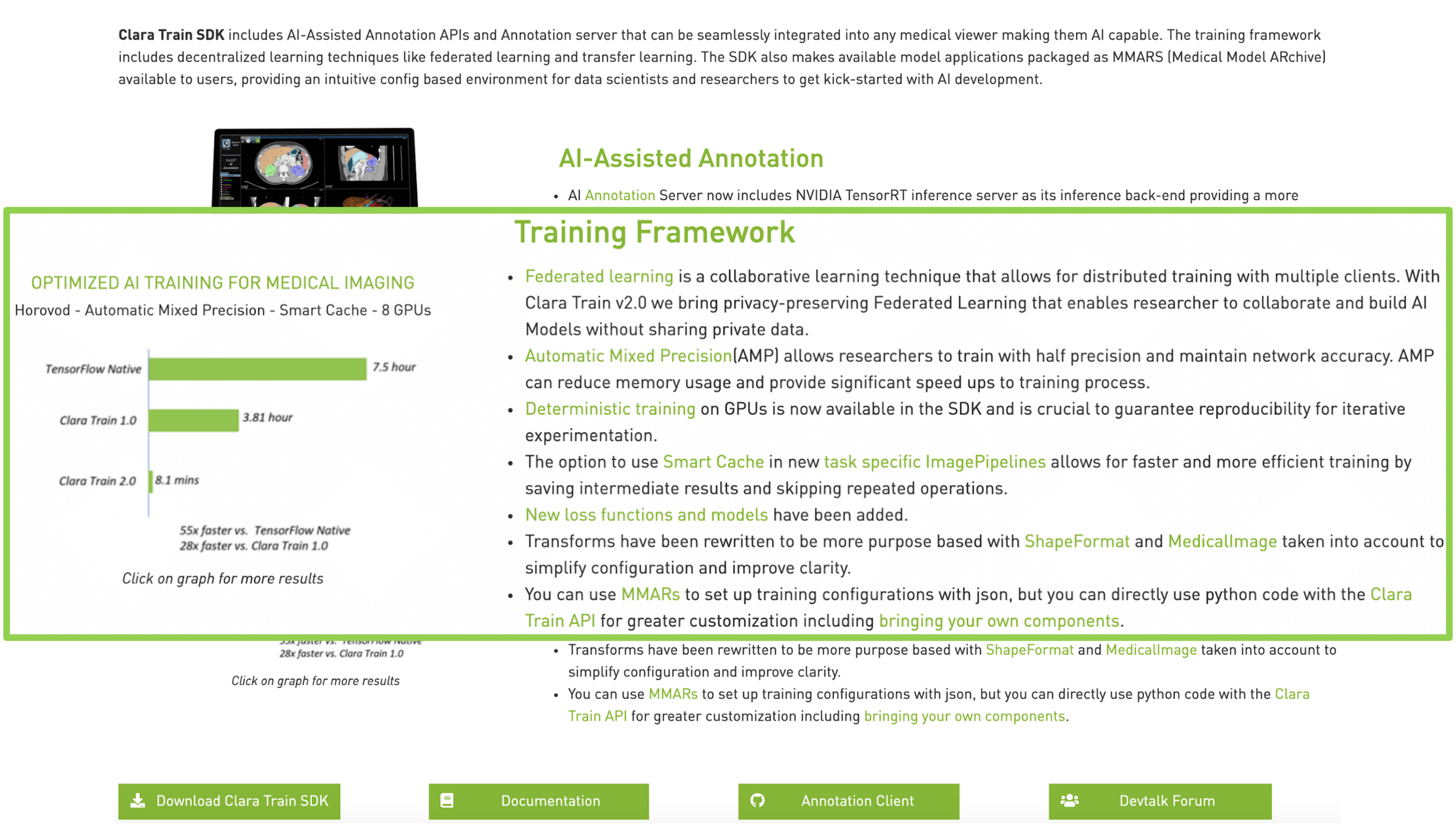
EXPORT

KUBERNETES

TESLA/QUADRO

NVIDIA DGX FAMILY

CLOUD



Clara Train SDK includes AI-Assisted Annotation APIs and Annotation server that can be seamlessly integrated into any medical viewer making them AI capable. The training framework includes decentralized learning techniques like federated learning and transfer learning. The SDK also makes available model applications packaged as MMARS (Medical Model ARchive) available to users, providing an intuitive config based environment for data scientists and researchers to get kick-started with AI development.

AI-Assisted Annotation

- AI Annotation Server now includes NVIDIA TensorRT inference server as its inference back-end providing a more

Training Framework

OPTIMIZED AI TRAINING FOR MEDICAL IMAGING
Horovod - Automatic Mixed Precision - Smart Cache - 8 GPUs
TensorFlow Native
7.5 hour
Clara Train 1.0
3.81 hour
Clara Train 2.0
8.1 mins
55x faster vs. TensorFlow Native
28x faster vs. Clara Train 1.0

Click on graph for more results

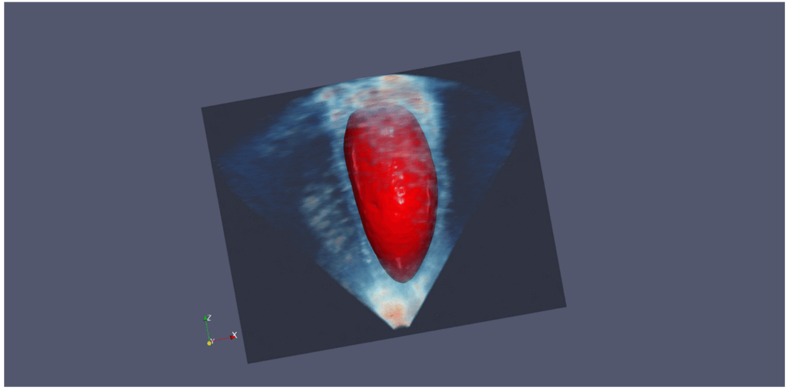
- Federated learning is a collaborative learning technique that allows for distributed training with multiple clients. With Clara Train v2.0 we bring privacy-preserving Federated Learning that enables researcher to collaborate and build AI Models without sharing private data.
- Automatic Mixed Precision(AMP) allows researchers to train with half precision and maintain network accuracy. AMP can reduce memory usage and provide significant speed ups to training process.
- Deterministic training on GPUs is now available in the SDK and is crucial to guarantee reproducibility for iterative experimentation.
- The option to use Smart Cache in new task specific ImagePipelines allows for faster and more efficient training by saving intermediate results and skipping repeated operations.
- New loss functions and models have been added.
- Transforms have been rewritten to be more purpose based with ShapeFormat and MedicalImage taken into account to simplify configuration and improve clarity.
- You can use MMARs to set up training configurations with json, but you can directly use python code with the Clara Train API for greater customization including bringing your own components.

55x faster vs. TensorFlow Native
28x faster vs. Clara Train 1.0

Click on graph for more results

- Transforms have been rewritten to be more purpose based with ShapeFormat and MedicalImage taken into account to simplify configuration and improve clarity.
- You can use MMARs to set up training configurations with json, but you can directly use python code with the Clara Train API for greater customization including bringing your own components.

https://developer.nvidia.com/clara-medical-imaging



Thank you!

Questions?