Universal Machine Learning for Topology Optimization

Heng Chi[a]

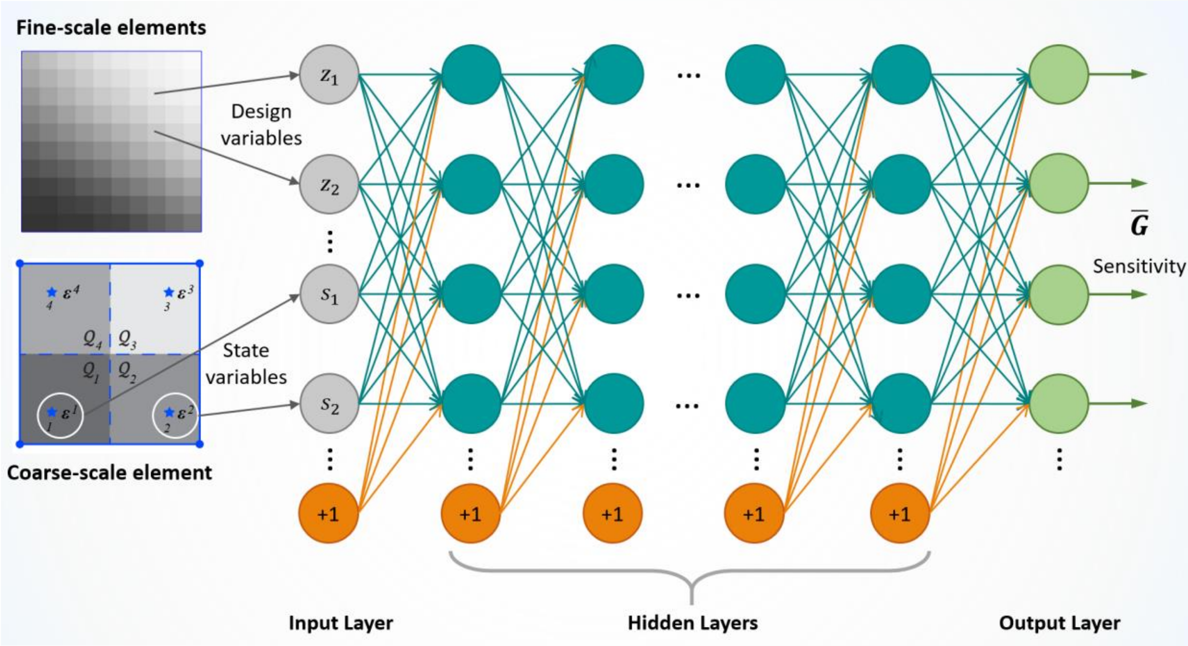
Yuyu Zhang[a]

Tsz Ling Elaine Tang[b]

Lucia Mirabella[b]

Livio Dalloro[b],

Glaucio H. Paulino[a]

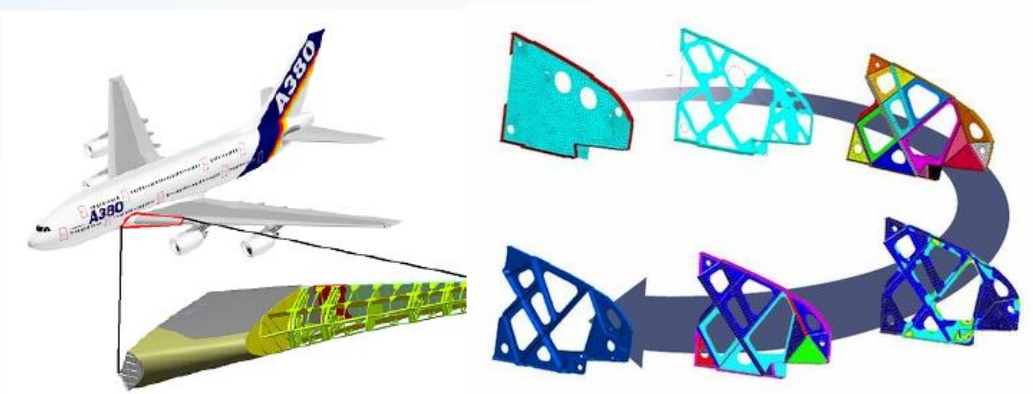


a: Georgia Institute of Technology

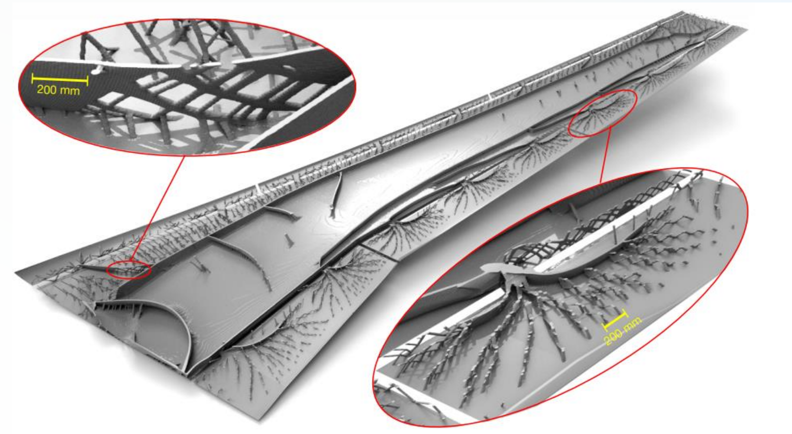
b: Siemens Corporation, Corporate Technology

SIEMENS

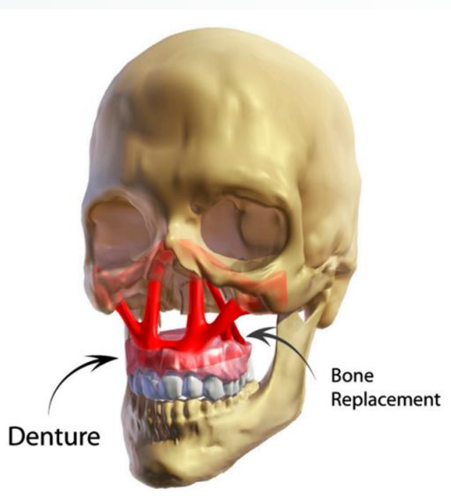
Topology Optimization in Engineering



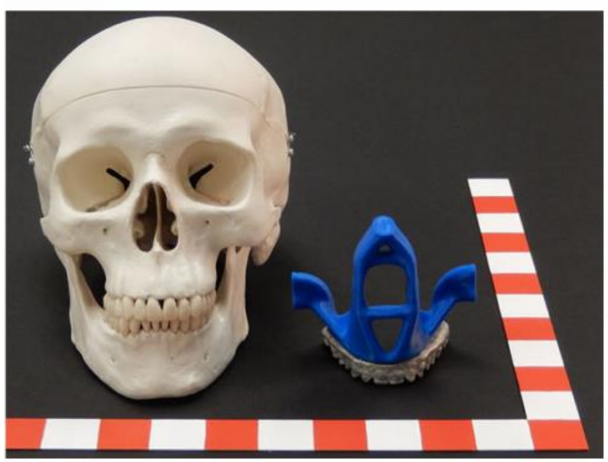
de Klerk and Sotirov, 2009



Aage, Andreassen, Lazarov, Sigmund, *Nature*, 2017



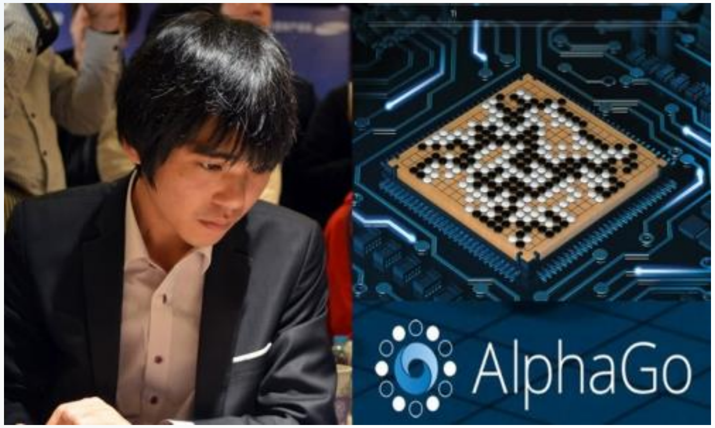
Sutradhar, Paulino, Miller, Nguyen, *PNAS*, 2010



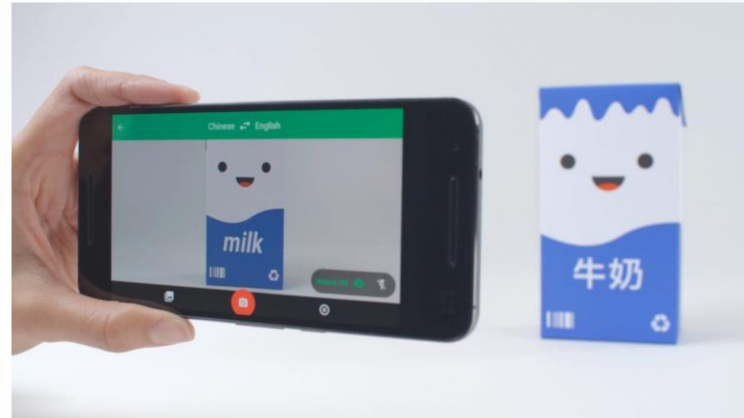
Zegard and Paulino, *JSMO*, 2016

3dprint.com

Machine learning and artificial intelligence



Go Game
(Silver et al. 2017)



Instant translation
(www.sciencemag.org)

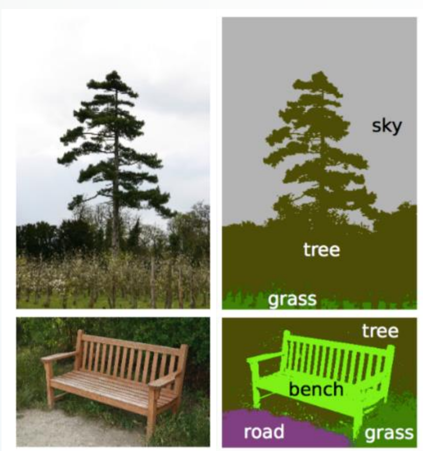
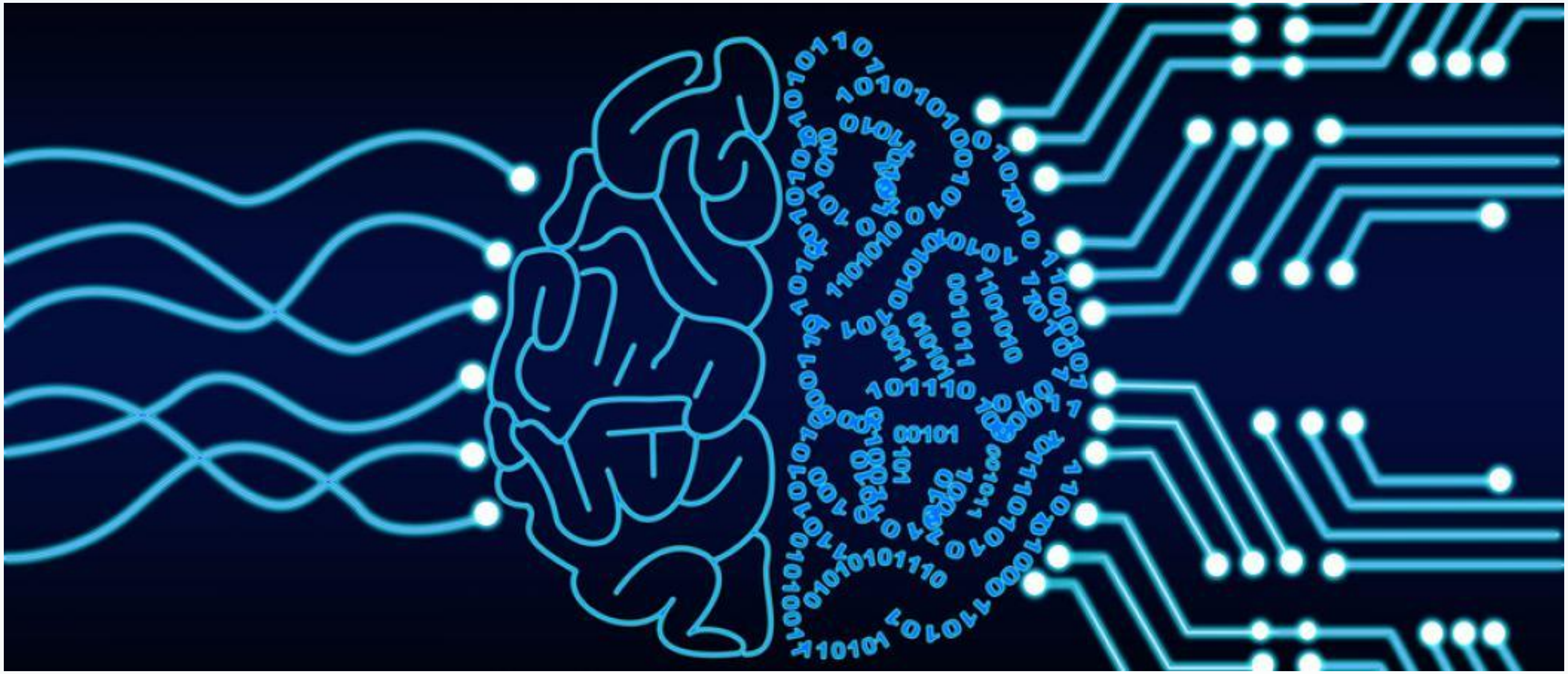


Image Segmentation
(Krähenbühl and Koltun, 2012)

Self-driving Car
(aitrends.com)

Artistic Style Transfer
(Andrychowicz et al. 2016)

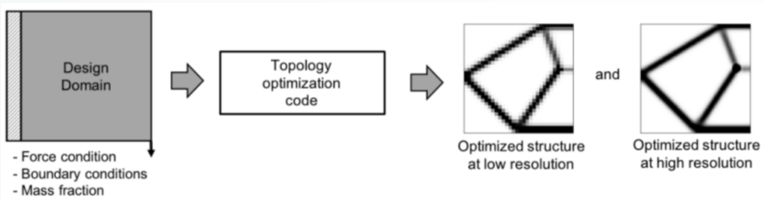


Can deep learning accelerate topology optimization without losing accuracy?

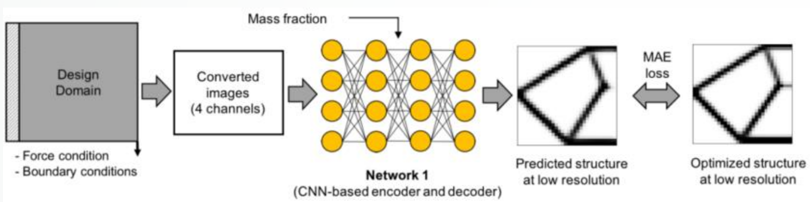
Machine learning in topology optimization

Training Stage

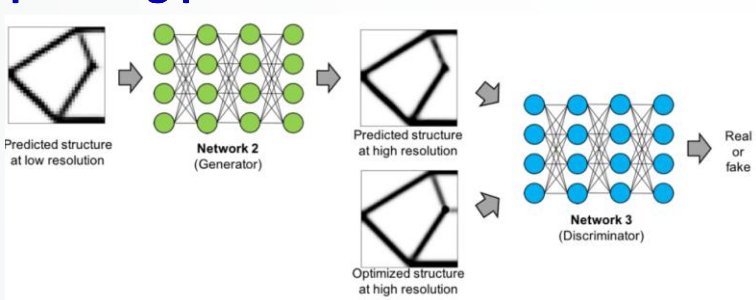
Step 1: generating dataset (with ~100,000 training data)



Step 2: training the CNN for low resolution prediction

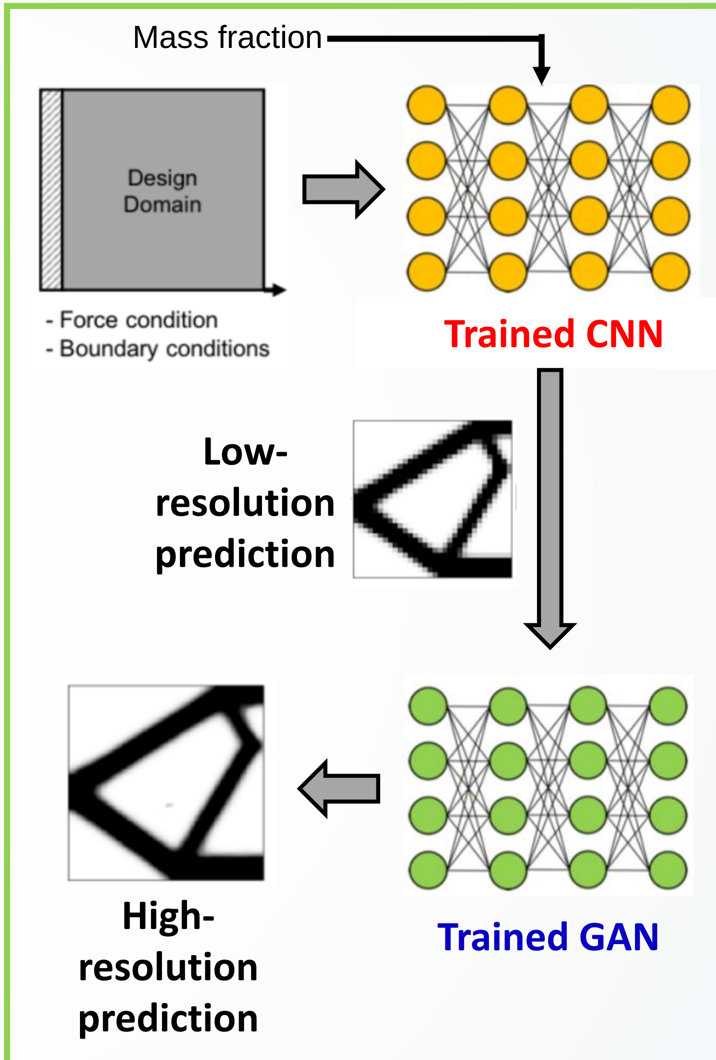


Step 3: training the conditional GAN for upscaling prediction



+

Prediction Stage

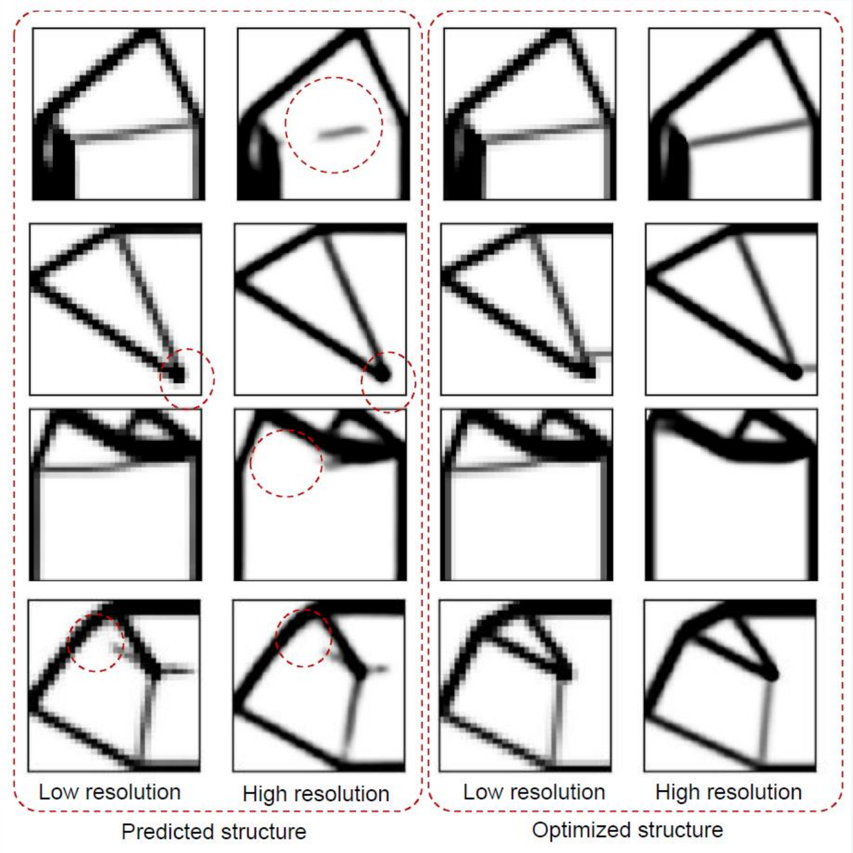


Y. Yu, T. Hur, J. Jung, and I. G. Jang. "Deep learning for determining a near-optimal topological design without any iteration", *Struct. Multidiscip. O.*, 2018

Limitations of the existing frameworks

Major Limitations:

- **Designs with structural defects**
- **Unable to perform large-scale designs**
- **Collecting training data is expensive**
- **Generalizable to any design domain?**



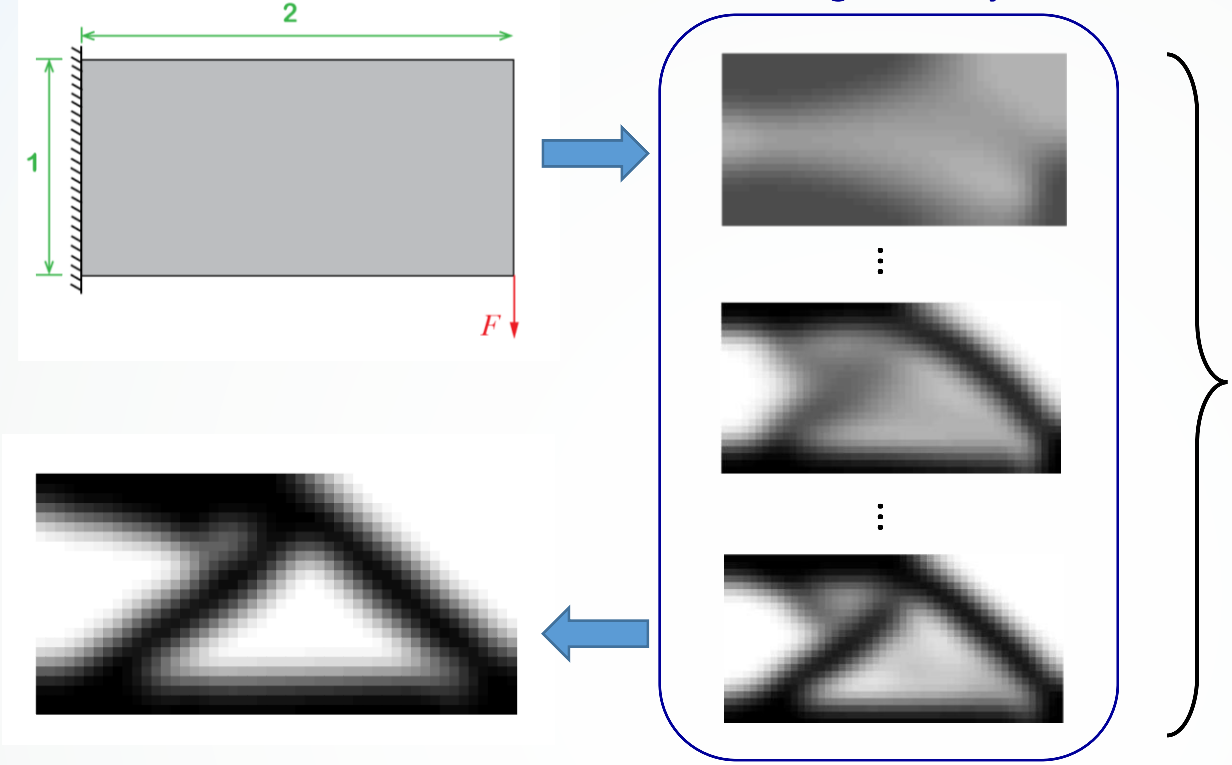
Yu, Hur, Jung and Jang, 2018

Training from history data of topology optimization

Design problem

Final design

Design history



Our idea:

To train a DL model based on the history data to learn the mapping between a given design and its sensitivity.

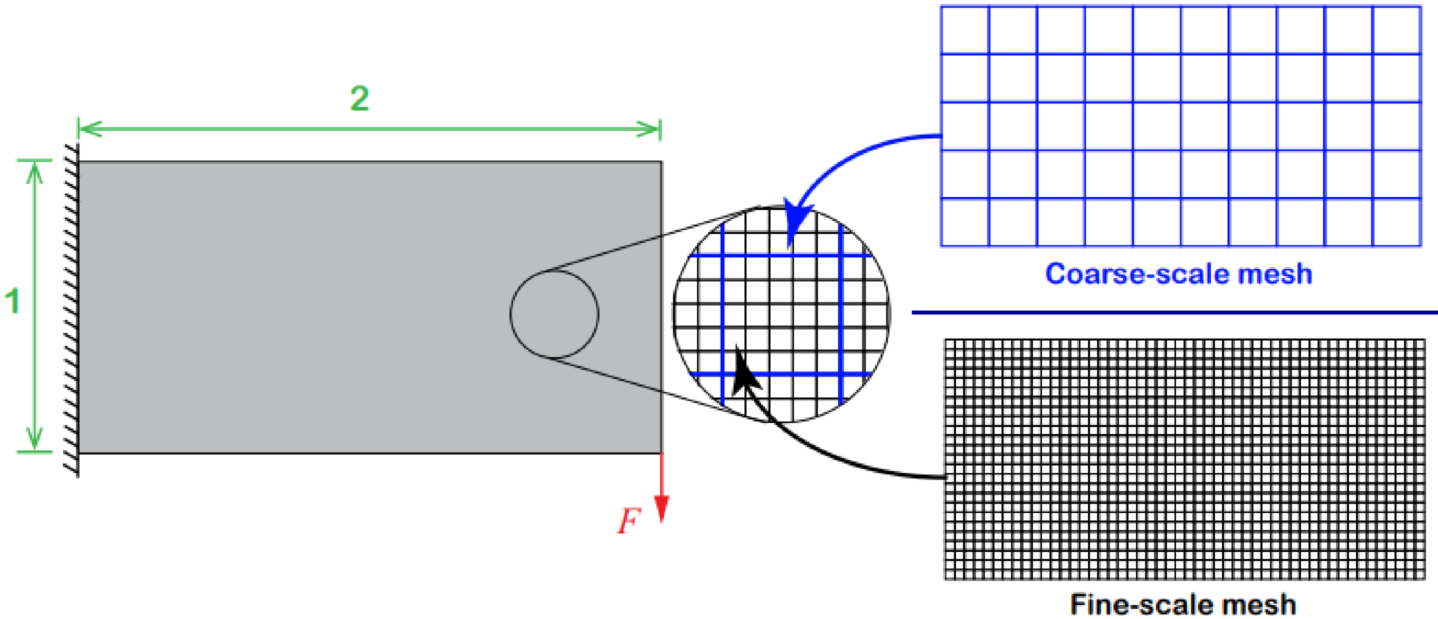
Key challenges:

1. Limited training samples
2. Unable to scale up
3. Limited model capacity

A Neural Network with 4 hidden layers and 1000 neuron per layer

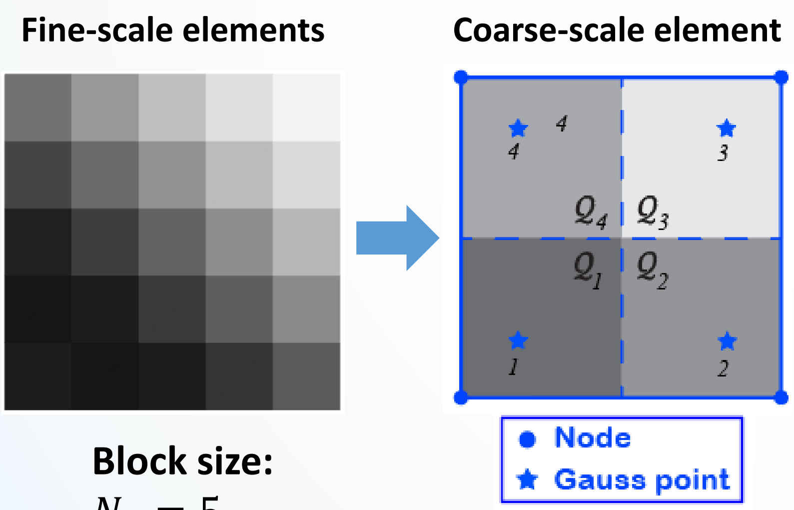
# of DVs	86K	250K	1.5M
# of params	175M	867M	3B
GPU memory	3.9GB	-	-

We propose a two-scale topology optimization setup



- State variable $\mathbf{u}^C$
- NO optimization

- Design variable $\boldsymbol{\rho}$
- Sensitivity
- Optimization



Block size:
$N_B = 5$

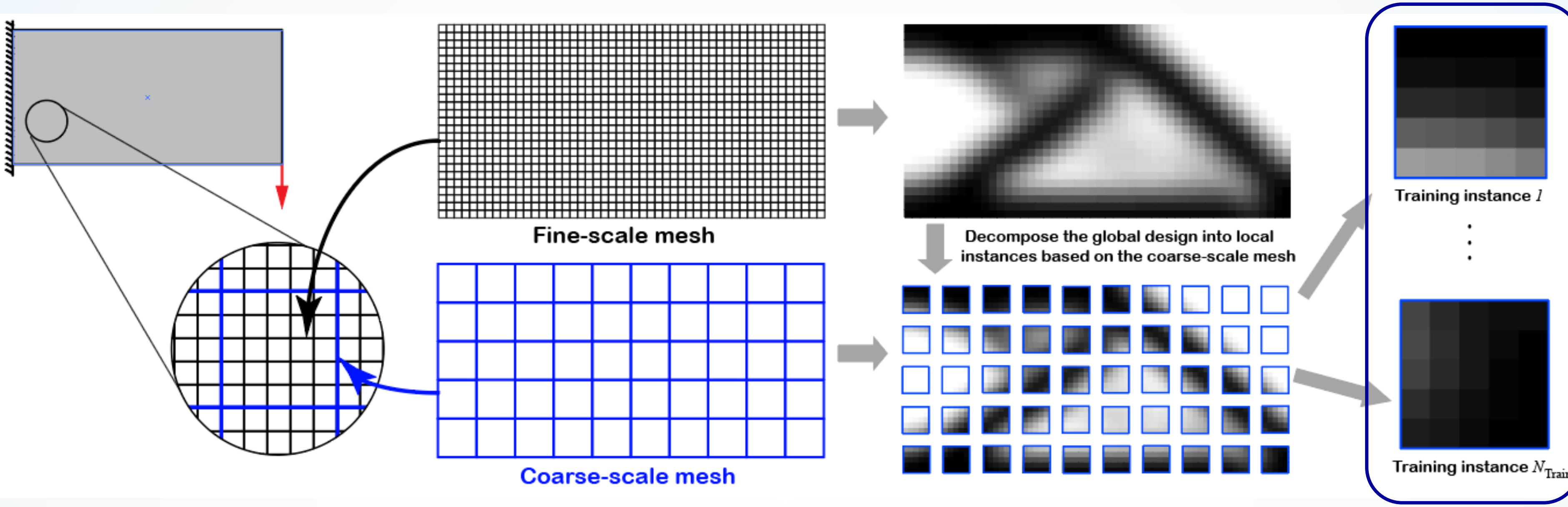
- The interpolated stiffness at jth integration point is defined as:

$$E_j^C = \frac{\sum_{i \in Q_j} w_i^{Q_j} E_i}{\sum_{i \in Q_j} w_i^{Q_j}}$$

- The local stiffness of a coarse-scale element is then computed as:

$$\mathbf{k}^C = \sum_j E_j^C W_j \left(\mathbf{B}_j\right)^T \mathbf{D}_0 \mathbf{B}_j$$

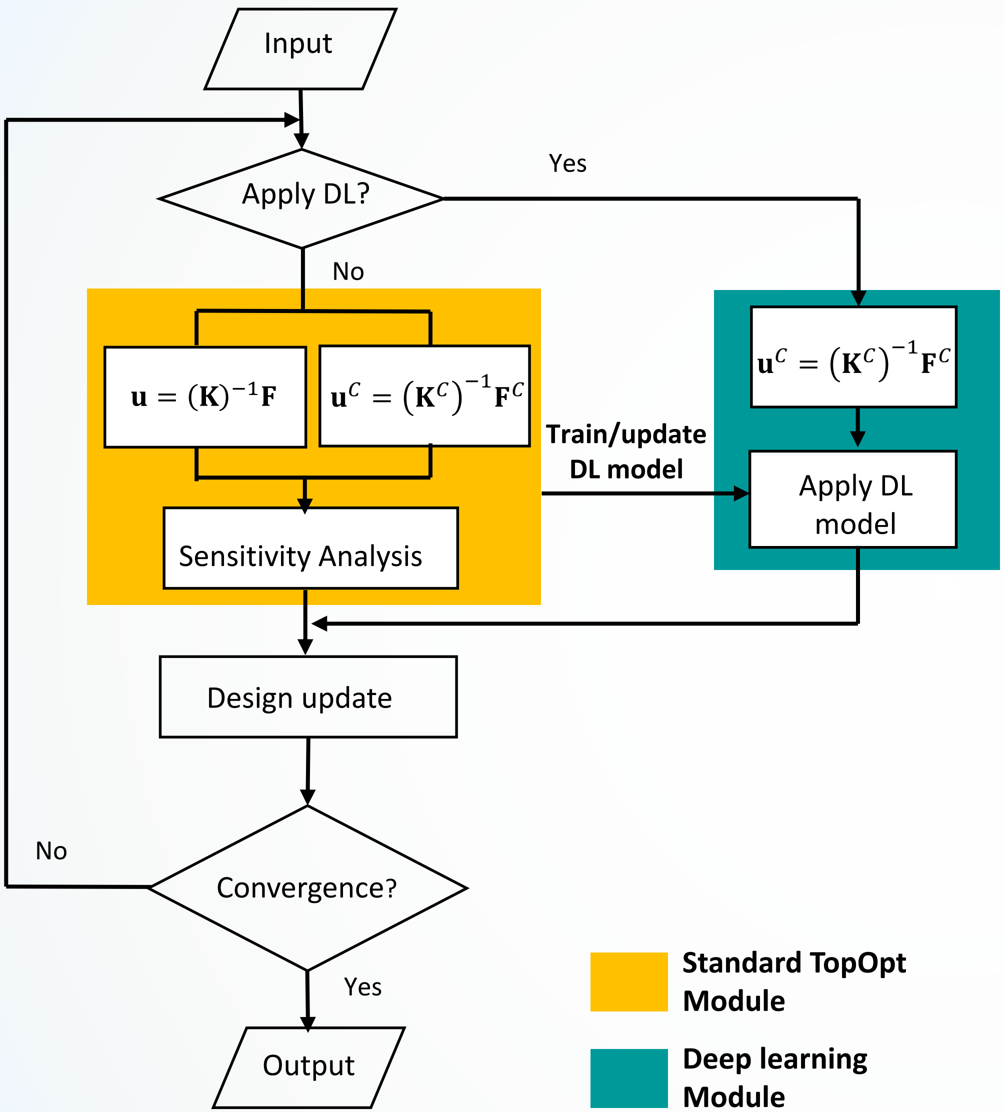
A tailored two-scale topology optimization setup



A Neural Network with 4 hidden layers and 1000 neuron per layer

	# of DVs	86K	250K	1.5M
Single-scale	# of params	175M	867M	3B
	GPU memory	3.9GB	-	-
Two-scale ($N_B = 5$)	# of params	3.3M	3.3M	3.3M
	GPU memory	0.7GB	0.7GB	0.9GB

Overall algorithmic flowchart



Main Features:

- No separate training step
- Online update to constantly provide new supervision
- Controllable GPU memory
- Highly scalable

Online Updating Scheme

Key Parameters:

N_I: Initial training step

N_F: Online update frequency

W_I: Initial training window size

W_U: Online update window size

An example of the online updating strategy with $N_I = 10, N_F = 10, W_I = 10, W_U = 5$:

Optimization steps as training/updating data

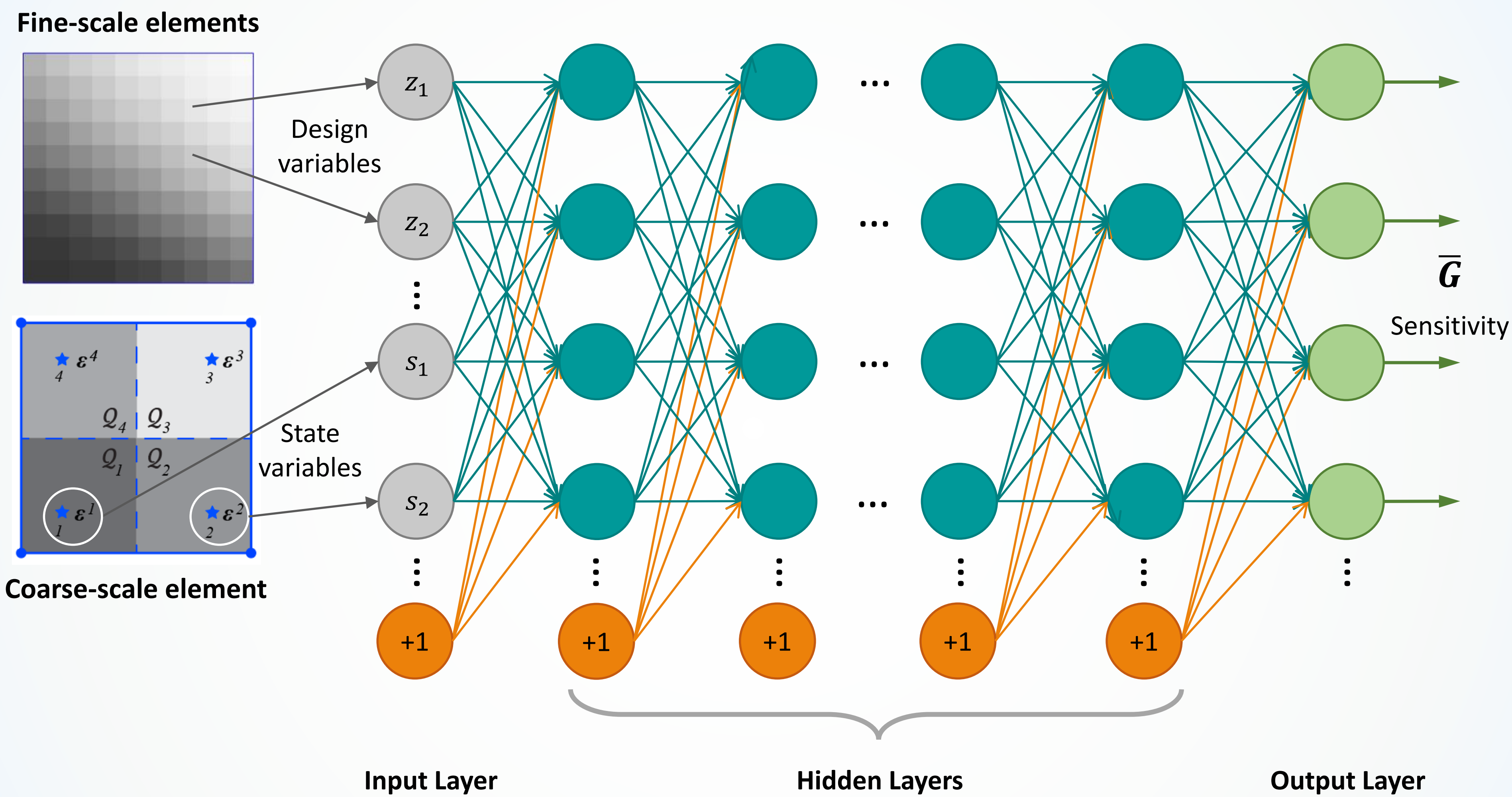
Initial training	11	12	13	14	15	16	17	18	19	20
1st Online updating	17	18	19	20	30					
2nd Online updating	18	19	20	30	40					
3rd Online updating	19	20	30	40	50					

⋮

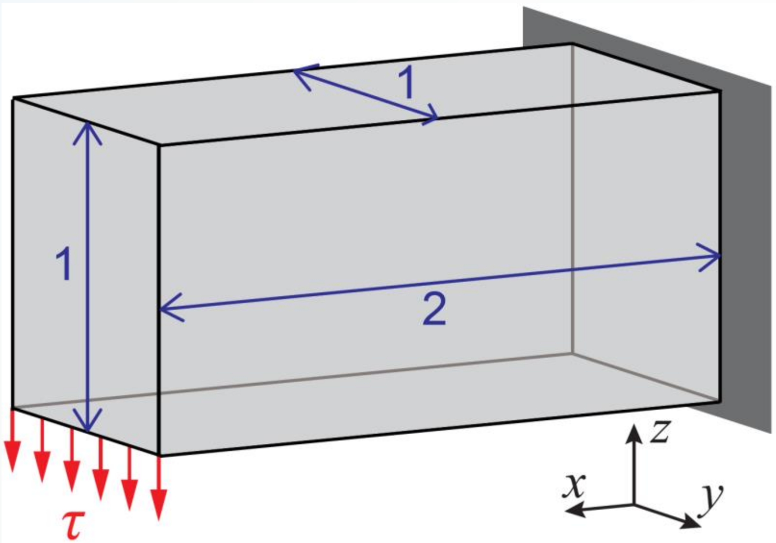
Initial training data

New training data

Architecture of the deep neural network



A cantilever design example



Design parameters:

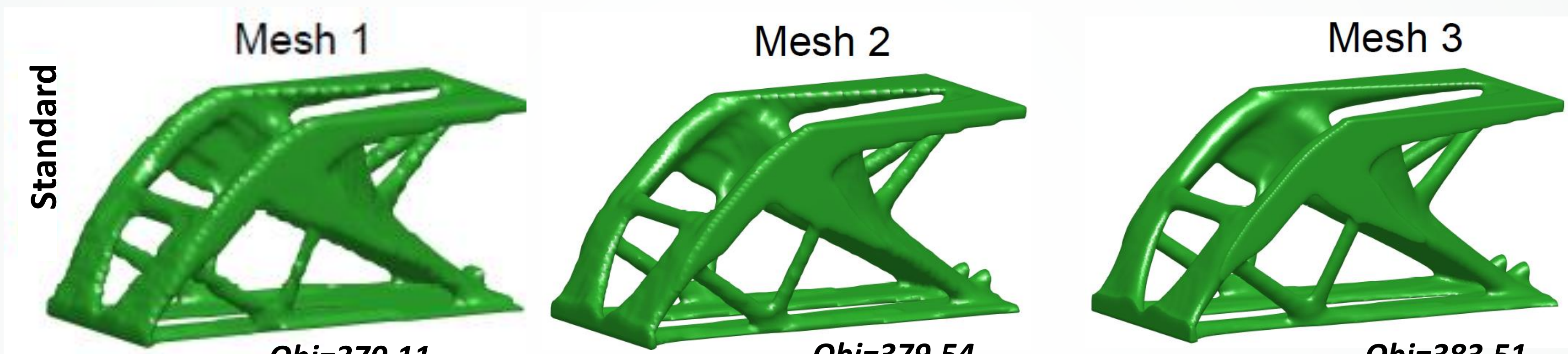
- Maximum optimization step: 200
- Volume Fraction: 12%
- $\tau = 1$
- Filter radius $R = 0.08$

$N_B = 5$	**Mesh 1**	**Mesh 2**	**Mesh 3**
# of DVs	86K	250K	1.5M
Fine-scale $\mathbf{K}$ size	276K	788K	4.5M
Coarse-scale $\mathbf{K}$ size	3K	8K	40K

The state equations are solved using PCG with Jacobi preconditioner on a *single* GPU.

Cantilever design: problem setup and final topologies

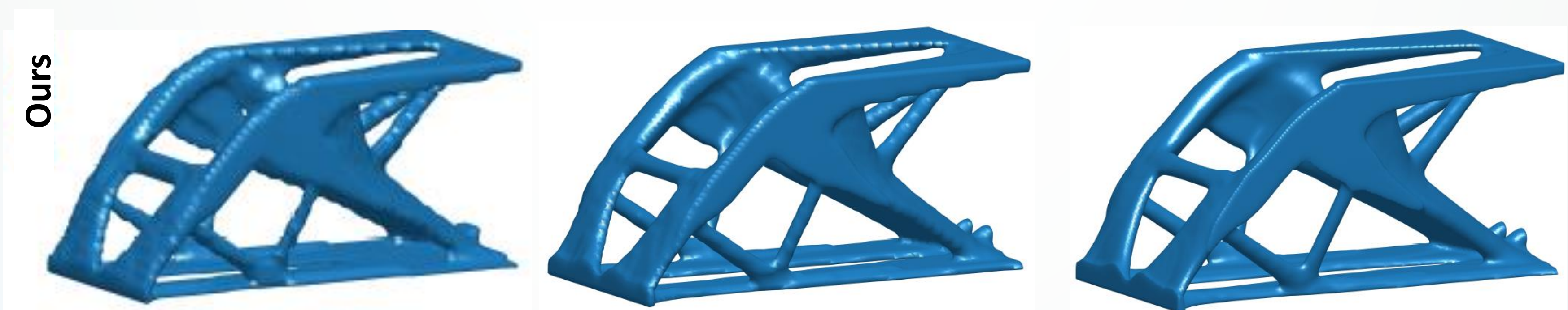
$N_I = 10$ (Initial training step)
$W_I = 10$ (Initial training window size)
$N_F = 25$ (Online update frequency)
$W_U = 2$ (Online update window size)



Obj=370.11
Exact solve: 200

Obj=379.54
Exact solve: 200

Obj=383.51
Exact solve: 200

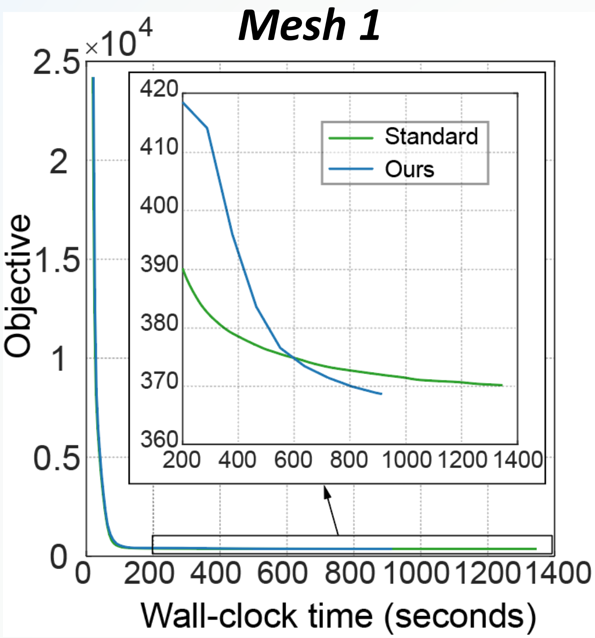


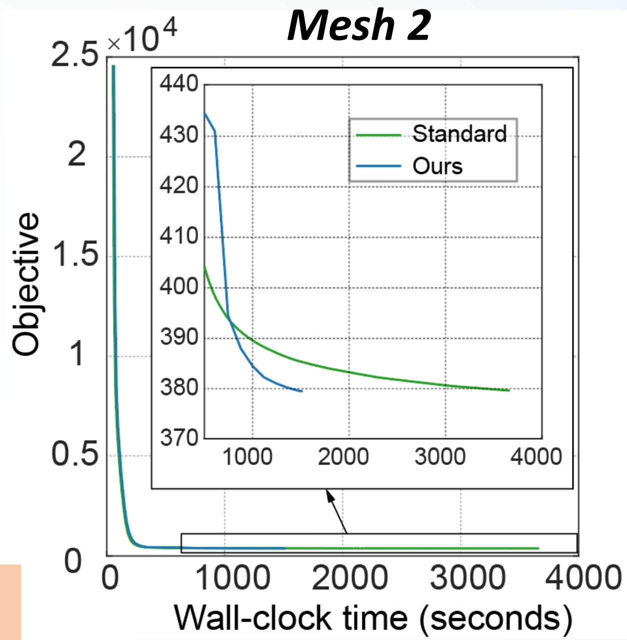
Obj=368.61
Exact solve: 27

Obj=379.37
Exact solve: 27

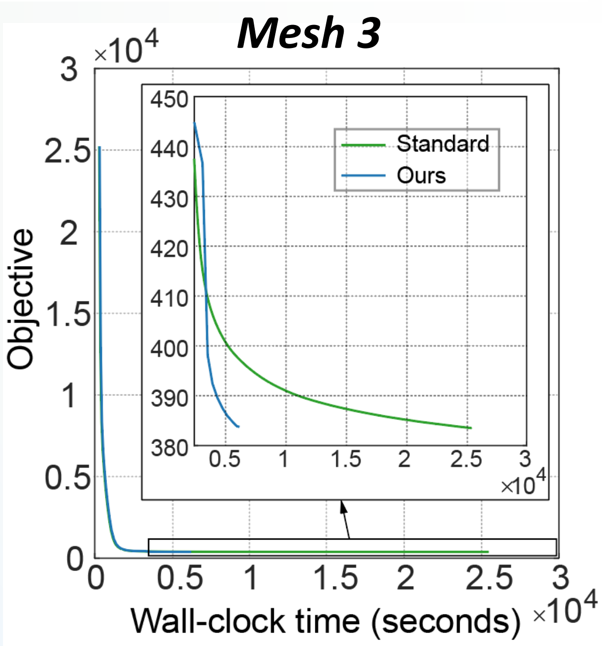
Obj=383.75
Exact solve: 27

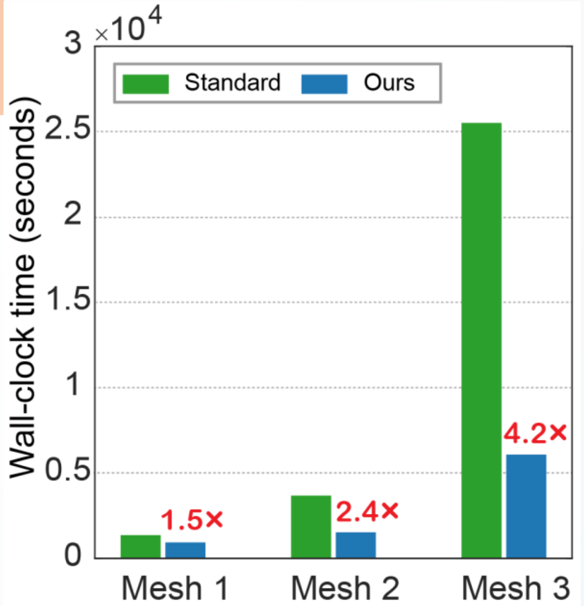
Cantilever design: convergence history and speedup



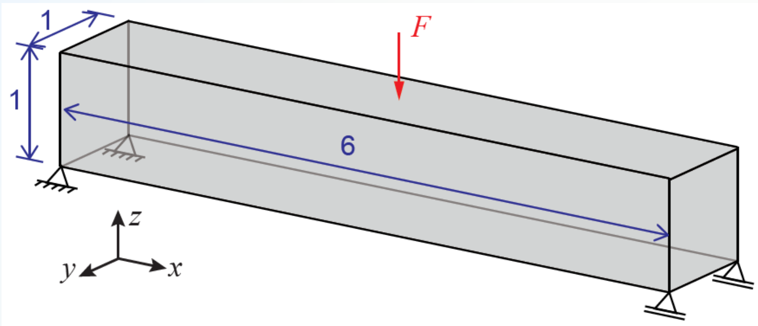


The larger the problem, the more speedup we get from our method.



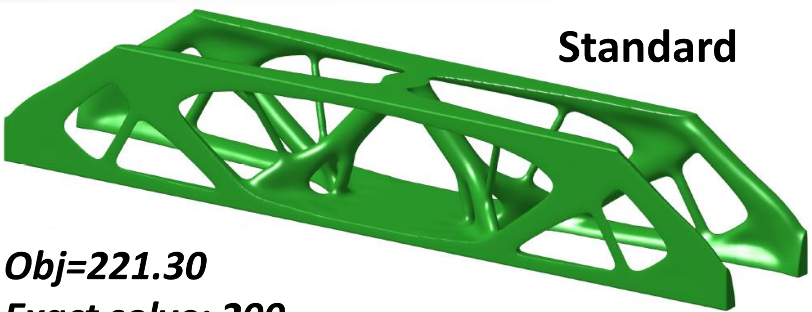


MBB beam design: problem setup and final topologies



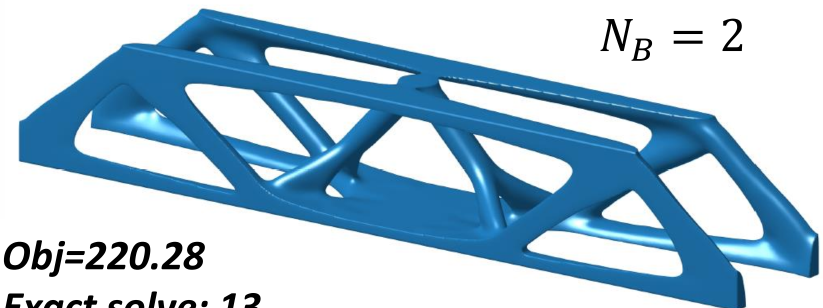
N_B	2	4	8
# of DVs	1.4M	1.4M	1.4M
Fine-scale $\mathbf{K}$ size	4.1M	4.1M	4.1M
Coarse-scale $\mathbf{K}$ size	533K	71K	10K

Standard



Obj=221.30
Exact solve: 200

$N_B = 2$

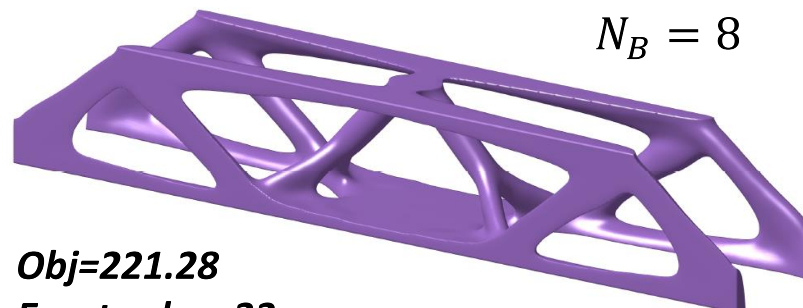


Obj=220.28
Exact solve: 13

$N_B = 4$

Obj=220.52
Exact solve: 14

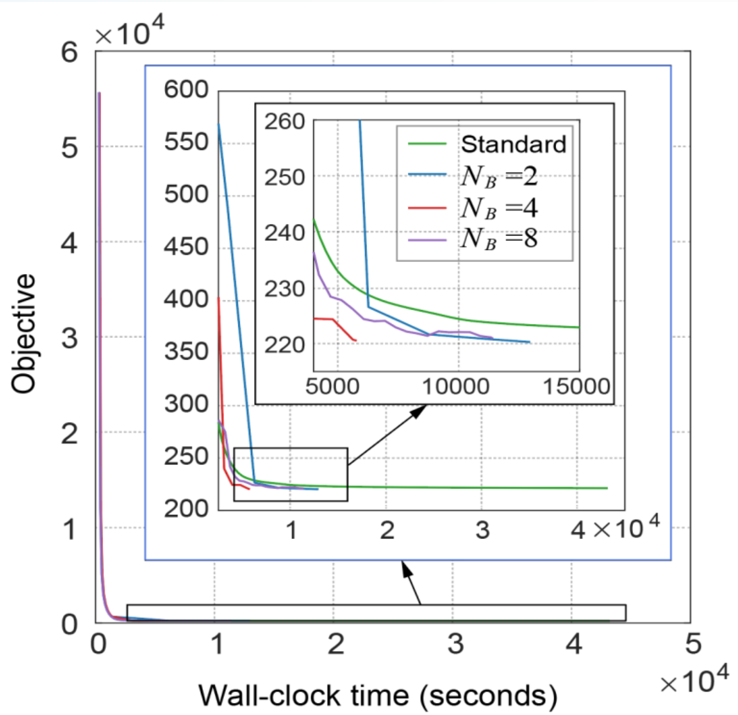
$N_B = 8$

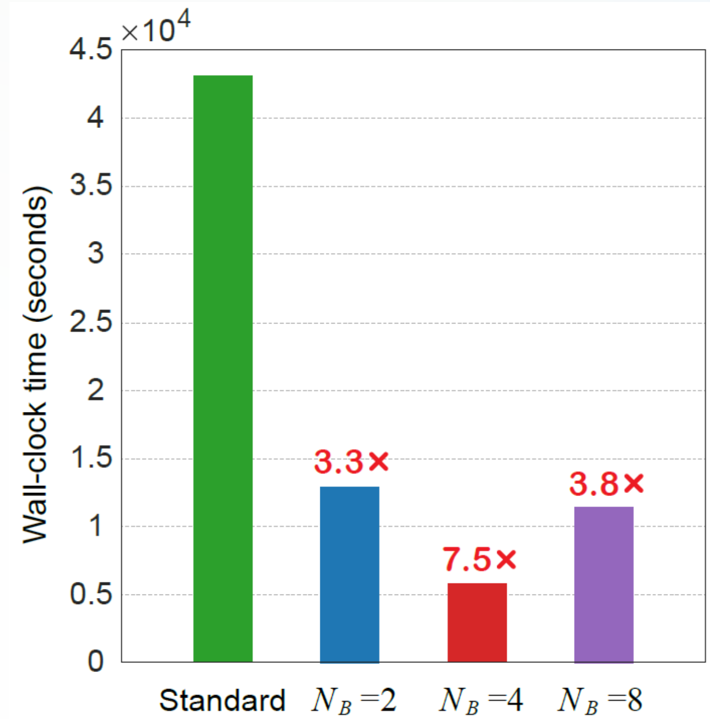


Obj=221.28
Exact solve: 33

Smaller block size allows us to provide less supervisions (exact solves) to the DL model in our proposed framework.

MBB beam design: convergence history and speedup





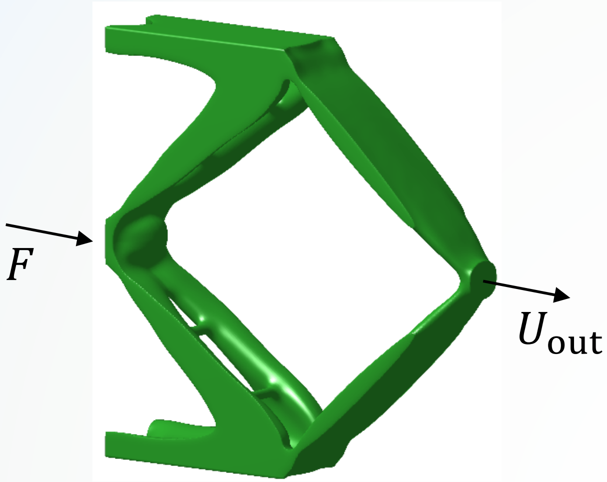
The trade-off between small and large block sizes:

- <u>**Smaller block sizes**</u>
  - ❖ Need less supervision→ less exact solves
  - ❖ Larger coarse-scale mesh
- <u>**Larger block sizes**</u>
  - ❖ Need more supervision→ more exact solves
  - ❖ Smaller coarse-scale mesh

Compliant mechanism design

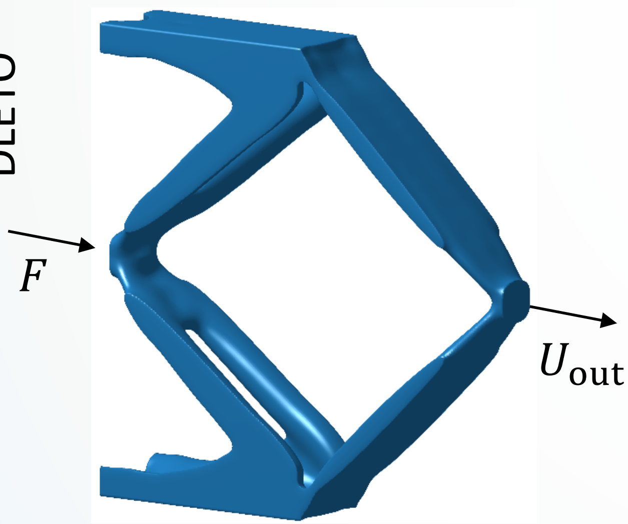
~1M design variables

Standard

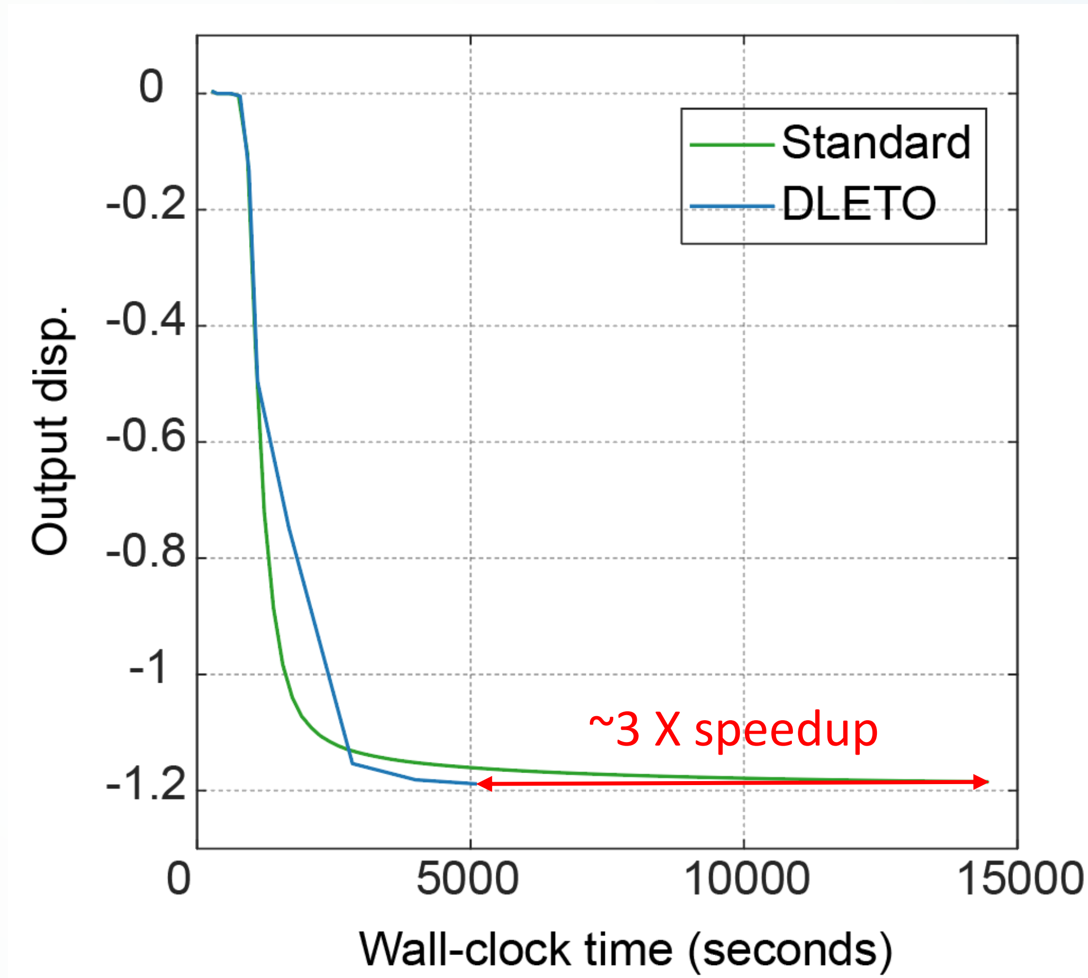


$U_{\mathrm{out}} = -1.185$, **100 exact steps**

DLETO



$U_{\mathrm{out}} = -1.189$, **12 exact steps**



Conclusions

- **We demonstrate that the proposed machine learning-based topology optimization framework is universal:**
  - ***No pre-collected training data is needed***
  - ***Can be readily applied to any design problems***
  - ***Can be potentially combined with any regression machine learning models***
- **The proposed machine-learning-based topology optimization framework can offer more speedup for problem of larger scale without any sacrifice in accuracy.**
- **With the two-scale topology optimization setup, the proposed framework is highly scalable and efficient.**