Accelerating CFD Simulations using Machine Learning

Super-Resolution: Coarse-to-Fine Resolution Simulations using Convolutional Neural network

DASSAULT SYSTEMES

Pradeep Singh, Nath Gopalaswamy, Francisco Martinez

ABSTRACT

Despite the progress in high performance computing, Computational Fluid Dynamics (CFD) simulations are still computationally expensive for many practical engineering applications such as simulating large computational domains and highly turbulent flows. One of the major reasons of the high expense of CFD is the need for a fine resolution to resolve phenomena at the relevant scale. The fine resolution requirements often drive the computational time, which makes long transient problems prohibitively expensive. In this on-going research project, we are proposing a convolutional neural network model that could learn end-to-end mapping between coarse resolution and corresponding high resolution simulation. This model is able to predict the high resolution simulation given its low resolution and could bring down the simulation time by a factor of 2 (or even more) at least.

Challenges in CFD

1. CFD simulations are know for memory usage and run times.
2. Need lot of computing resources.
3. Takes lot of time to converge.
4. Data generated is not being used.

What can we do within the Machine Learning framework?

1. Build ML models that could approximate the CFD solvers.
2. Use ML Models as a good initial guess for CFD solvers.
3. Embedded ML model in CFD solvers.
4. Make ML model compete with CFD solvers.

What is Image Super-Resolution?

- ❑ Reconstruct a High-Resolution image from observed Low-Resolution images.
- ❑ Algorithm uses a training samples to learn the fine details of an image at low-resolution.
- ❑ It then uses those learned relationships/ mappings to predict fine details in other images.

Our Approach

- ❑ Build a CNN based model that could learn end-to-end mapping between low-resolution and corresponding high-resolution simulations.
- ❑ Then, given a flow simulation, our model should be able to predict the high-resolution version of that same simulation.

- ❑ Assumption: Coarse resolution simulation includes all the information required for learning how simulation will evolve from coarse to fine resolution in time.
- ❑ Our model will directly learn this evolution in time and will be able to map the coarse resolution to fine resolution.

Dataset

1. Dataset is consist of 100 training and 5 testing samples.
2. XFlow is used to generated high-resolution simulation images.
3. NACA 4412 Airfoli is simulated using Xflow.
4. We changed two parameters to generate training samples:
   - Angle of attack (AOA): -10 to 10 (degrees)
   - Velocity value: 10 to 50 (m/sec)

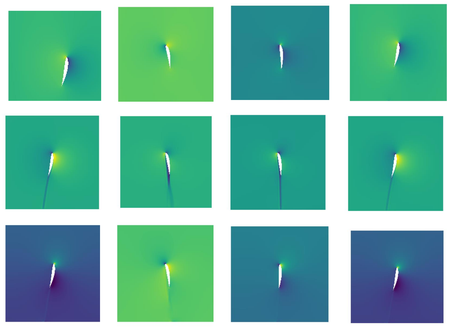


Fig. 1: This figure shows NACA 4412 simulation using Xflow. P, Vx and Vy are shown in first, second and third row respectively.

Preparing Data for Training and Testing:

1. To prepare training data, we first downsample the original training images by the desired scaling factor n to form the LR images. Then we crop the LR training images into a set of f_{sub} x f_{sub}-pixel sub-images with a stride k.
2. The corresponding HR sub-images (with size $(nf_{sub})^2$) are cropped from ground truth images.
3. These LR/HR sub-images pair are the primary training data.

Models: Architecture

1. SRCNN: Super Resolution using Convolutional Neural Network
2. FSRCNN: Faster Super Resolution using Convolutional Neural Network

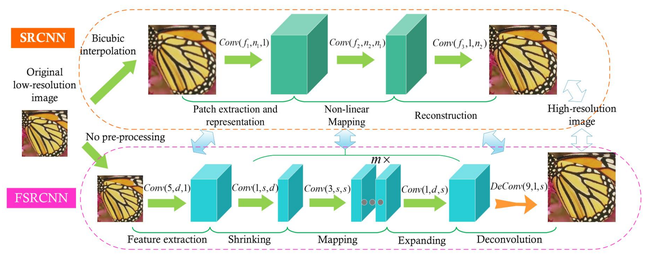


Fig. 2. This figure shows the network structures of the SRCNN and FSRCNN.

SRCNN:

1. Preprocessing: Upscale given low-resolution image to the desired size using bicubic interpolation.
2. Patch extraction and representation: Extracts (overlapping) patches from low-resolution image and represents each patch as a high dimensional vector.
3. Non-linear mapping: Nonlinearly maps each high-dimensional vector onto another high-dimensional vector.
   - Each mapped vector is conceptually the representation of a high-resolution image.
4. Reconstruction: Aggregates the above high-resolution patch wise representation to generate final high-resolution image. This image is expected to be similar to ground truth.

FSRCNN:

1. Feature Extraction: Perform feature extraction using convolutional layer and represent each extracted patch as a high-dimensional vector.
2. Shrinking: Usually extracted features dimensions is very large. Shrinking layer reduces the feature dimensions.
3. Non-linear Mapping: Maps each high-dimensional vector onto another high-dimensional vector.
   - Multiple mapping layers are added.
4. Expanding: Inverse of shrinking layer.
5. Deconvolution: Upsamples and aggregates the previous features with a set of deconvolution filters.

Difference between SRCNN & FSRCNN architecture:

1. FSRCNN adopts the original low-resolution image as input without bicubic interpolation unlike SRCNN.
2. The non-linear mapping step in SRCNN is replaced by three steps in FSRCNN, namely the shrinking, mapping, and expanding step.
3. FSRCNN adopts smaller filter sizes and a deeper network structure.

These improvements provide FSRCNN with better performance but lower computational cost than SRCNN.

Training Models

Training Time for SRCNN Model

Samples	Type	Epochs	Time (hours)
90	Grey	100	> 8
90	Grey	500	48
10	Color	100	4-6
10	Color	1000	> 24

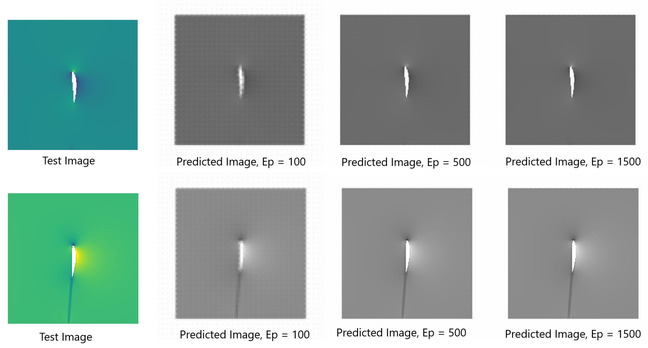
Training Time for FSRCNN Model

Samples	Type	Epochs	Time (hours)
90	Grey	100	3-5
90	Grey	500	20-24

Fig. 3. Above you can see training time for SRCNN and FSRCNN with different number of training samples and training cycles. Both models were trained on same system and infrastructure.

Results

Below you can see results from our FSRCNN model which was trained on our in-house dataset generated using XFlow.



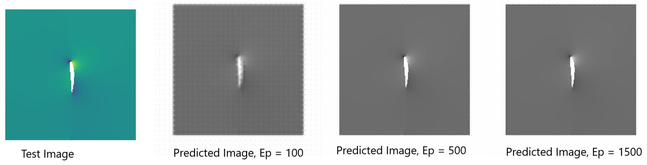


Fig. 4. This figure shows the results from FSRCNN model. The first image (left most) is a test image from Test set and rest three images are predicted by our model with different training cycles (or as we call it Epochs). First row is of Pressure field P, second is of velocity field Vx in x direction and last row is of velocity field Vy in y direction.

Conclusion

- ❑ Our model is able to learn mapping between low and high resolution simulations from our dataset.
- ❑ As of now, we can at least upsample a low-resolution simulation by a factor of x 2, x 3 and x 4 for a given model trained on same set of LR/HR pairs.

Ongoing Work

- Incorporate 3D geometry.
- Test or work with different geometry/ shapes.
- Make model general enough that it could be able to learn mapping between different shapes/ geometries.
- Could we increase scaling factor by factor of x 8 or more?
- Real-time Super resolution could be a possibility.

REFERENCES:

1. Image Super-Resolution Using Deep Convolutional Networks - arXiv:1501.00092
2. Accelerating the Super-Resolution Convolutional Neural Network - arXiv:1608.00367
3. Deep CNN with Residual Net, Skip Connection and Network in Network - arXiv:1707.05425
4. Single Image Super-Resolution using an efficient Sub-Pixel Convolutional Neural Network - arXiv:1609.05158
5. Single Image Super-Resolution using a Generative Adversarial Network - arXiv:1609:04802
6. Super-resolution reconstruction of turbulent flows with machine learning - arXiv:1811.11328
7. Super-Resolution via Deep Learning - arXiv:1706.09077
8. Machine Learning for Fluid Dynamics - arXiv:1905.11075

ACKNOWLEDGEMENTS

I would like to thanks Ruddy BRIONNAUD for providing support in generating dataset using XFlow and the entire CFD R&D team for their valuable suggestions during discussions/ team meetings. Finally this work wouldn't have been possible without the support and guidance of my managers Nath GOPALASWAMY and Francisco Martinez.