Patch-Wise Training with Convolutional Neural Networks to Synthetically Upscale CFD Simulations

John P. Romano

Alec C. Brodeur

Oktay Baysal

AIAA SciTech Forum
23-27 January 2023, National Harbor, MD & Online
AIAA SCITECH 2023 Forum

10.2514/6.2023-1804

Patch-wise Training with Convolutional Neural Networks to Synthetically Upscale CFD Simulations

John P. Romano* and Alec C. Brodeur†
Naval Surface Warfare Center, Dahlgren Division, Dahlgren, Virginia, 22448

Oktay Baysal‡
Old Dominion University, Norfolk, Virginia, 23529

This paper expands the authors' prior work[1], which focuses on developing a convolutional neural network (CNN) model capable of mapping time-averaged, unsteady Reynold's-averaged Navier-Stokes (URANS) simulations to higher resolution results informed by time-averaged detached eddy simulations (DES). The authors present improvements over the prior CNN auto-encoder model that result from hyperparameter optimization, increased data set augmentation through the adoption of a patch-wise training approach, and the predictions of primitive variables rather than vorticity magnitude. The training of the CNN model developed in this study uses the same URANS and DES simulations of a transonic flow around several NACA 4-digit airfoils at high angles of attack[1]. The authors test the updated model by inputting airfoil profiles and flow conditions outside of the training set and by comparing the output flow field against DES calculations. The Fully Unstructured Navier-Stokes 3D (FUN3D) solver from NACA generates the computational fluid dynamics (CFD) simulations and uses computing assets available from the Department of Defense High Performance Computing Modernization Program (HPCMP) and Old Dominion University (ODU) High Performance Computing (HPC). Finally, the paper includes the effects of these techniques on the predictive capability and the performance of the authors' CNN model.

I. Nomenclature

α	=	angle of attack	`elu`	=	exponential linear unit
ρ	=	density	`mse`	=	mean squared error
c	=	chord	`mae`	=	mean average error
Ma	=	Mach number	`mape`	=	mean average percentage error
`N`	=	generic patch size	`msle`	=	mean squared logarithmic error
p	=	pressure	`relu`	=	rectified linear unit
Re	=	Reynolds number	`selu`	=	scaled exponential linear unit
s	=	scaling factor			
x	=	horizontal Cartesian grid coordinate			
y	=	vertical Cartesian grid coordinate			

II. Introduction

Over the past five years, researchers have put significant effort and investment in leveraging machine learning (ML) to provide improved predictive capability to existing CFD simulations [2] and to measure fluid phenomena more accurately [3–5]. This has become a natural technique enabled by increases in computational power, a corresponding increase in the prevalence of high-fidelity simulation techniques within the field of CFD, and the use of scale-resolving simulations and direct numerical simulations (DNS). The application of ML to CFD problems uses (1)

*Head, Hypersonic Design, Integration & Analysis Branch, Member, AIAA
†Aerospace Engineer, Threat Engineering Branch
‡Professor & Eminent Scholar, Mechanical & Aerospace Engineering, obaysal@odu.edu, Associate Fellow, AIAA

increased simulation fidelity by augmenting solution algorithms and physical models (e.g., turbulence modeling and physics-informed ML) and (2) shorter solution runtimes using reduced-order modeling and super-resolution.

One of the more popular approaches to algorithm and model augmentation uses inverse modeling, where ML algorithms adapt parameters within existing turbulence models to create models with improved generalizability. Researchers at the University of Michigan [6–10] have studied these techniques in depth over the past several years, where inverse modeling infers updated functional forms of and addresses deficiencies with popular turbulence models. This work results in enhanced predictive capability for separated wake flow and other problem-of-interest simulations. Researchers at NASA Langley investigate the efficacy of these techniques for separating flows over airfoils [11]. Pochampalli (et al., [12]) estimates correction fields for the Spalart-Allmaras turbulence model by considering CNN architectures.

Reduced-order and super-resolution models aim to learn high-resolution representations of lower resolution flow fields by leveraging high-resolution CFD, specifically Detached Eddy Simulation (DES), Large Eddy Simulation (LES) and DNS, and experimental data. Several research groups over the past several years have undertaken related studies in reduced-order and super-resolution modeling techniques. Agostini [13] develops a CNN auto-encoder model with the goal of optimal data compression. Fukami (et al., [14]) and Matsuo (et al., [15]) investigate CNN and "Downsampled Skip-Connection Multi-Scale Models" to perform super-resolution reconstruction of turbulent flow fields. Lui and Wolf [16] build surrogate models with CNNs to predict future flow-field states in periodic transient flow cases. Kochov (et al., [17]) introduces a blended approach to leveraging ML for improved CFD computation by considering ML-augmented solver paradigms, inverse learning, and ML-augmented interpolation functions to improve the computational efficiency of subgrid scale models for LES calculations.

In the present study, the authors discuss several techniques for improving the the CNN auto-encoder model presented at the 2022 SciTech Forum [1]: hyperparameter optimization, patch-wise model training, and encoding airfoil geometry within the CNN model as a separate input layer. These techniques provide incremental improvements to the prior model, so the study compares sample results from the SciTech 2022 model with the results from the newly generated models. The subsequent sections qualitatively describe the predictive performance of the newly presented models by considering contour plots of pressure and density fields generated by URANS calculations, DES calculations, and various CNN models. The paper quantitatively compares CNN model predictions against DES truth data by including plots of training loss history. In [1], the use of a CNN model as a *post-processing step* after URANS calculations reduces computational cost by roughly 100 times compared to performing DES calculations for the same flow conditions for the airfoil cases of interest. The new model modifications focus on the CNN *pre-processing step* of formatting URANS data into a form that the CNN model can interpret, and on adjustments to training processes. These modifications do not result in a significant change in the computational resource requirements for CNN model generation, training, or predictions; therefore, the cost analysis from [1] still holds. Consequently, this paper does not discuss computational resource requirements or improved efficiency resulting from CNN model prediction use.

III. Theory

Several CNN U-net models in this paper extend the models of [1] and reflect rewrites that focus on modularity and the removal of a bottleneck layer to ease parametric model shape studies. The previous study develops a CNN auto-encoder model to predict relatively high-accuracy flow field (based on time-averaged DES) from relatively low-accuracy flow field (based on URANS, time-averaged with the same scheme as used for DES). This work uses the same NASA FUN3D CFD solver (version 13.1)-generated [18] data sets from [1]. Python scripts using Tensorflow (version 2.2.0) generate the ML models and perform data processing. A hyperparameter optimization study using the Sequential Model-based optimization for general Algorithm Configuration (SMAC) [19, 20] python library informs optimal network shape and sizing.

Table 1 outlines the search space for hyperparameter optimization, provides the baseline parameters from [1], and presents SMAC-generated optimal parameter sets. Within this search space, individual convolution blocks consist of batch normalization, convolution, and max pooling layers; de-convolution blocks consist of up-sampling and de-convolution layers. In building a balanced U-net, the number of convolution blocks must equal the number of de-convolution blocks in the model. All other hyperparameters within the trade space represent direct user parameters within Tensorflow-implemented Keras framework. For completeness, Table 1 gives the latent dimension value of the baseline model [1] (i.e., the number of neurons within the previously presented auto-encoder bottleneck layer); however, the new networks of this paper reflect the removal of the bottleneck layer with the move from an auto-encoder network to a U-net. As a point of comparison, Fig. 1 shows the baseline CNN auto-encoder model from [1]; Fig. 2 depicts the new, hyperparameter-optimized U-net architecture.

Hyperparameter	***Type***	***Search Space***	***Baseline Model [1]***	***Optimal Model***	***Optimal Masked Model***
Convolution Filters	Uniformly distributed integer value	[10 : 100]	48	8	15
Activation Function	Categorical	`sigmoid` `relu` `elu` `tanh` `selu`	`sigmoid`	`relu`	`sigmoid`
Filter & Pooling Kernel Size	Categorical	[2, 4, 8]	2	2	4
Number of Convolution & Deconvolution Layers	Uniformly distributed integer value	[1 : 4]	3	4	2
Network Optimizer	Categorical	`adam` `adadelta` `adagrad` `adamax` `nadam`	`adadelta`	`adam`	`nadam`
Loss Function	Categorical	`mse` `mae` `mape` `msle`	`msle`	`mse`	`mase`
Latent Dimension	N/A	N/A	12	N/A	N/A

Table 1 Hyperparameter Optimization Search Space and Optimal Network Configuration

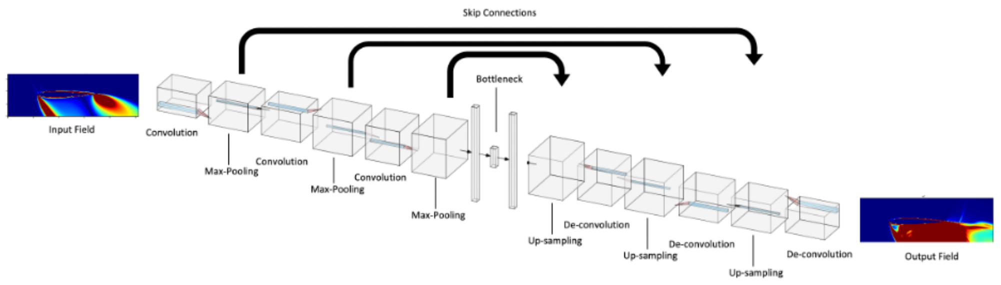

Fig. 1 Baseline SciTech 2022 Network Layout

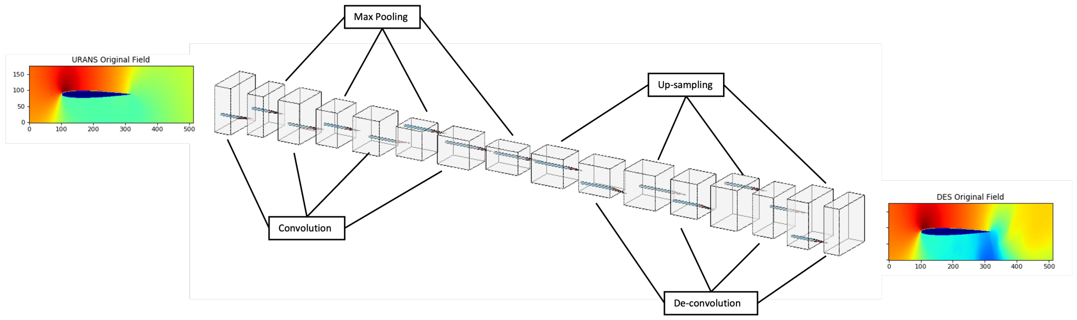

Fig. 2 Optimized Model Network Layout

The present study investigates various data pre-processing and training techniques to understand how to generate the best-performing CNN model (i.e., it results in increased URANS simulation accuracy). It uses the base data sets from the previous work: four NACA 4-digit airfoil profiles across a range of angles of attack (α), as listed in Table 2. All simulations occur at Mach 0.72 and Re = 12.5E6. URANS and DES simulation results for these conditions pre-process through several steps to create data sets ready to feed through the CNN models. As a proof of concept, the new model uses only the angle of attack sensitivity data set. (For completeness, Table 2 includes all data from [1]).

Fig. 3 depicts the general pre-processing procedure. First, a cubic interpolation function places a region of the flow field that immediately surrounds the airfoil surface onto a uniformly distributed Cartesian grid to replicate the structure of an image file. Fig. 3 outlines this in red. Equation 1 governs the size of the Cartesian grid. These Cartesian grid representations of the simulation data then break down into many `N x N` regions by placing patches, with `N` denoting the patch width and length in number of pixels, within the pre-processed domain and considering only the data within the patch region. Fig. 3 represents this with the orange region. This paper studies two methods of placing these patches: (1) systematically sweeping the patch region across rows within the Cartesian regions and (2) sampling the Cartesian regions by randomly placing the patch region origin within the Cartesian regions. In the random sampling approach, *NumPy* random number generators determine the (x, y) coordinates for the bottom right corner of the patch region in each patch example. Regardless of patch method, a one-to-one mapping between the input URANS data and the reference DES data must exist after patch sampling. This requires the concurrent sampling of patches within the URANS and DES data to ensure that the inputs and outputs of the CNN model pair properly during training. The study applies this patch-wise sampling approach to both the training and the testing data. Some of the presented results look at the performance of the model to predict updated flow-field data without using patch-decomposed data at model test time. The inclusion of these results allow comparison with the results that use patch-decomposed testing data; an additional post-processing step after CNN model processing then recombines these data. The fields of image processing, medical imaging [21], other fluid and thermal transport studies [22, 23] have used these patch-decomposition techniques.

Study	***Training Data Set***	***Testing Data Set***
α Sensitivity	NACA0006 $\alpha \in [-30°, -10°, 0°, 10°, 30°]$	NACA0006 $\alpha \in [-20°, 20°, 25°]$
Geometry Sensitivity	NACA0006 $\alpha \in [20°, 30°]$ NACA0012 $\alpha \in [20°, 30°]$ NACA4412 $\alpha \in [20°, 30°]$	NACA0006 $\alpha = 25°$ NACA0012 $\alpha = 25°$ NACA4412 $\alpha = 25°$ NACA2412 $\alpha \in [20°, 25°, 30°]$

Table 2 Description of Data Sets [1]

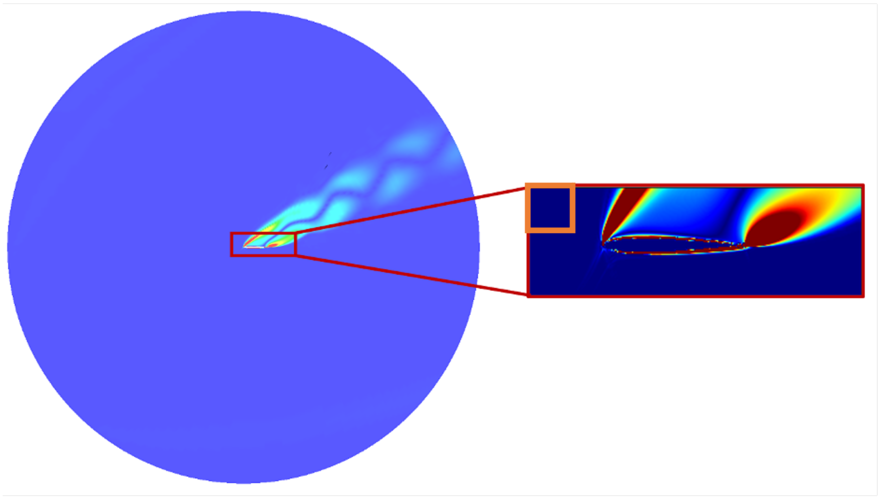

Fig. 3 Pre-processing Grid Transformation Schematic

$$\begin{aligned} x &\in [-0.1sc, 1.1sc] \\ y &\in [-0.4sc, 0.4sc] \\ s &\in [1.00, 1.05, 1.10, 1.15, 1.20, 1.25] \end{aligned} \tag{1}$$

The authors use several methods to combined the results obtained from the patch-decomposed data sets to determine the optimal performance, investigating two methods for the systematically patched data sets and one method for the randomly patched data sets. For the systematic patched data, the first method passes only coincident patch regions through the trained models at testing time to predict new flow fields. In this case, each point in the Cartesian grid, except for those on the patch boundaries, is present only in one patch region. Overlapping data between patches are present only at the boundary between two patches. This approach requires no further data to form a complete prediction of new flow-field data. In the second method for the systematic patched data, the testing data patches pass through the trained model to predict new flow-field data, with overlapping data regions existing between many patches. For grid locations in areas where several patches overlap, the calculation of an average field variable generates a complete new flow-field prediction. With random patched data, no data reconstruction occurs because the captures does not include the patch locations during flow-field decomposition. Instead, the full pre-processing test data pass through the U-net models trained with random patches. Consequently, the random patch method performed the worst of the methods considered.

IV. Results

The authors developed many CNN models in executing this study by considering multiple sized data set patches (`32x32`, `64x64` and `128x128` grid points in width and height, respectively) and different pre-processing techniques described above. The settings and structures within each of these models are identical and set as Table 1 describes, and the patch size and subsequent data sets for training and testing use are the only variable parameters. The models with a geometry-masking layer use a different set of settings than the single input layer models; however, a new SMAC optimization squeezes the most performance out of the multiple input layer model. Table 1 also gives the optimized parameter set of the model with the geometry-masking layer. Table 2 describes the training data sets each model uses during 10,000 training epochs. For the models that used randomly distributed, patch-decomposed training data, 5000 random patches were generated for the `32x32`, `64x64` and `128x128` patch sizes. Table 3 defines a nomenclature to differentiate the different models and to reference model results in this section more easily. Each symbol in Table 3 describes a single model parameter that can vary, and several symbols combine to form a single model definition. For example, `PSCC32` denotes model results representing a U-net model that trains on pressure field data and decomposes into `32x32` grid-point-sized patches using the systematic, row-wise technique that Section III describes. This result would present test data that decompose into patches systematically and combine using only coincident patches while neglecting patches that overlap with one another. In this nomenclature, one symbol is selected from each of the following sets: `{P,D}`, `{S,R}`, `{32,64,128}` and `{F,CC,CA}`. The symbol denoting inclusion of the geometry-masking layer, `M`, is used when the additional mask input layer is present but neglected for single input field models. The model with mask layer did not include patch decomposition, so models including the `M` symbol do not denote a patch size. All results denoted URANS and DES represent data coming from the input URANS simulations and reference DES simulations, respectively.

Nomenclature Parameter	*Model Component*	*Nomenclature Parameter*	*Model Component*
`P`	Pressure field input/output	`S`	Systematic patching
`D`	Density field input/output	`R`	Random patching
`32`	`32x32` patch size	`128`	`128x128` patch size
`64`	`64x64` patch size	`256`	`256x256` patch size
`F`	Test data passed through trained model as full fields, not patch-decomposed	`CC`	Test data patch-decomposed and combined during post-processing considering only coincident patches
`M`	Geometry masking layer input/output	`CA`	Test data patch-decomposed and combined during post-processing considering all patches

Table 3 U-net Model Result Nomenclature

Figs. 4-8 show representative nondimensional pressure and density result comparisons for the various patch sizes and patch techniques that this study explores. These comparisons seek to determine qualitatively the pre-processing process that produces the best performing U-net model. In each of these figures, each column represents a testing case; the left column represents $\alpha = -20°$, the middle row represents $\alpha = 20°$, and the right row represents $\alpha = 25°$. Each row represents the results from a particular model: the top row always represents the input URANS field data, and the last row always represents the reference DES field data that the U-net model attempts to emulate. The intermediate rows differ depending on the models under consideration, and these follow the model nomenclature of Table 3. Figs. 4 and 7 look at pressure and density field predictions, respectively, for the three different patch pre/post-processing techniques. The random patch predictions generally are of lower quality than the systematic patch predictions, and the two different post-processing techniques (coincident and average recombination) tend to produce similar answers. However, some subtle differences are detectable. In the pressure results, the averaged reconstruction produces a smoother field than the coincident reconstruction. This is particularly evident in the $\alpha = 20°$ case where the separation region above the airfoil exhibits fewer and finer striations in the averaged reconstructed field than in the coincident reconstructed field. For the density results, the average and coincident reconstructions are almost identical. However, the implementation of the average reconstruction results in some erroneous density values in the spots that did not occur in the coincident reconstruction.

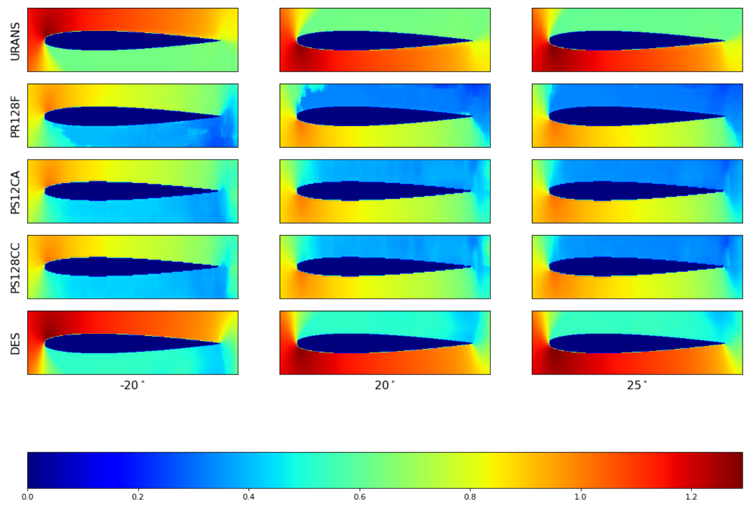

Fig. 4 Patch-wise Training Non-dimensional Pressure Result; Comparing Patch Technique for NACA 0006 Airfoil and Patch Size 128x128

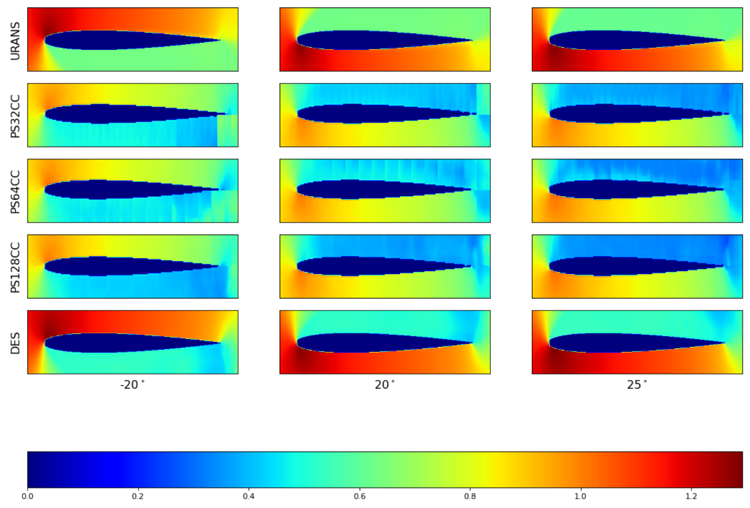

Fig. 5 Patch-wise Training Non-dimensional Pressure Result; Comparing Patch Size for NACA 0006 Airfoil and Coincident Patch Reconstruction

Figs. 5 and 8 compare the pressure and density predictions, respectively, for the three different patch sizes and for the best performing pre- and post-processing techniques, systematic patching with coincident reconstruction. Fig. 5

makes apparent the prediction differences among models trained on different patch sizes. This figure shows that as the patch size increases, the number of discontinuities within the flow-field reconstruction decreases, and the spacing between the discontinuities increases.

In the case of the `128x128` patches, the discontinuities almost completely resolve out of the flow-field reconstruction. The accuracy of the reconstruction, using the DES data as the truth, also increases with increasing patch size, but none of the results fully captures the correct flow-field features. In particular, pressure gradients are less extreme in the reconstructed flow fields than in the URANS and DES flow fields. Moreoever, lambda wave formations probably exist in the CFD data that ML model reconstructions fail to capture properly. In the density results, Fig. 8, the differences between each ML model result are minimal. Upon close inspection, there are differences in the $\alpha = 25°$ reconstructions in the low-density region immediately above the airfoil trailing edge. This region tends to grow with increasing patch size, with the `128x128` reconstruction most closely matching the DES reference data, but with the reconstruction not producing a large enough low-density region to capture fully the correct flow-field behavior. The pressure and density reconstructions exhibit a similar behavior when using the averaged patch recombination technique as seen with the coincident patch recombination technique.

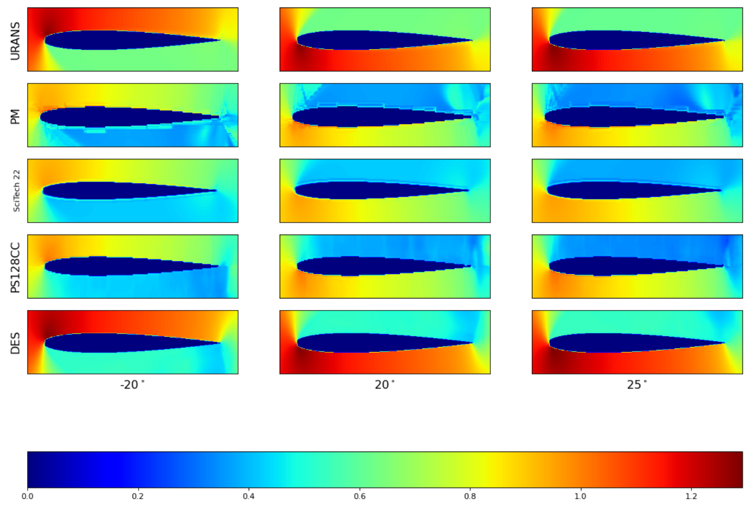

Fig. 6 Patch-wise Training Non-dimensional Pressure Result; Comparing Best Performing Patch and Masked Data for NACA 0006 Airfoil

Fig. 6 compares the pressure reconstructions among the models containing an additional geometry mask layer with the best performing `128x128` trained model (`PS128CC`). The inclusion of the geometry-masking layer seeks to improve the ability of the ML model to replicate correctly the airfoil geometry from the input URANS data. In previous iterations of the patch-trained models, model reconstructions repeatedly show trouble with correctly replicating the leading and training edge geometry. Including the geometry-masking layer does correct this issue, but further improvements to the patch-trained model, such as preforming a refined hyperparameter optimization and increasing the number of training epochs, also improve the geometry reconstruction across all cases. However, the authors include the geometry-masking layer to show a representative case not using patch-training was not used and for completeness. As a point of comparison, Fig. 6 includes the baseline pressure prediction results coming from the prior model [1]. The model with the geometry-masking layer predicts erroneous flow-field features within the separation regions. In general, the patch-trained model that uses coincident data combination performed smoother and more accurate pressure field predictions. These predictions, however, tend to exhibit lower pressure values throughout the flow field than the input RANS or the DES data that the model should emulate.

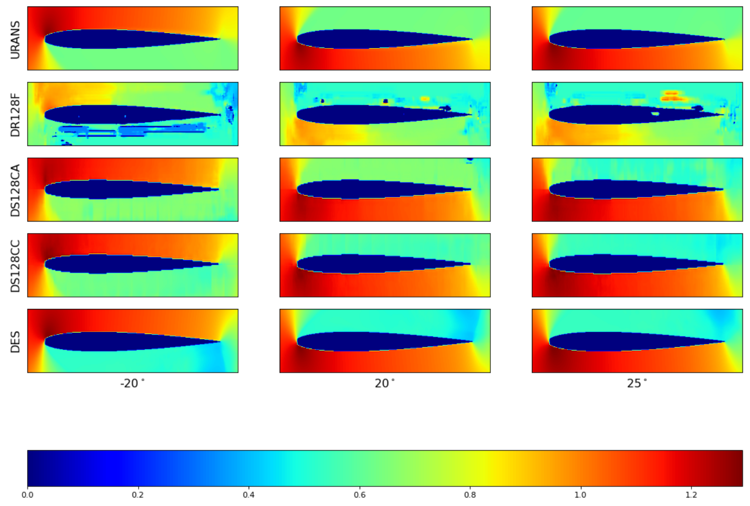

Fig. 7 Patch-wise Training Non-dimensional Density Result; Comparing Patch Technique for NACA 0006 Airfoil and Patch Size 128x128

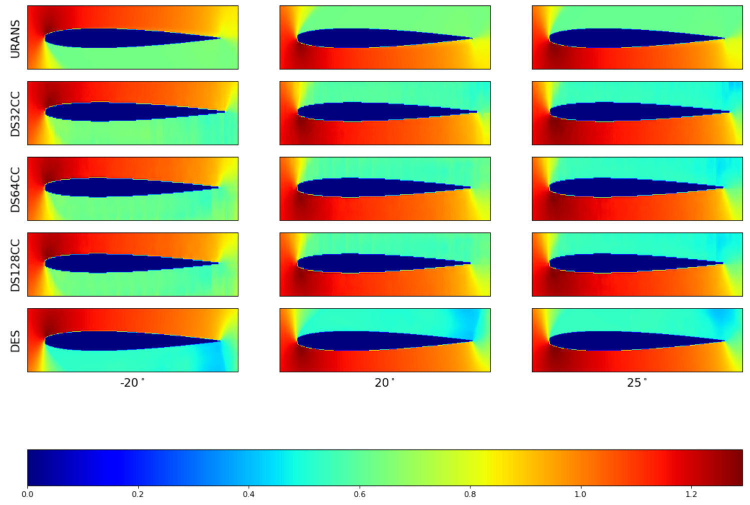

Fig. 8 Patch-wise Training Non-dimensional Density Result; Comparing Patch Size for NACA 0006 Airfoil and Coincident Patch Reconstruction

Figs. 9-11 show the training loss histories for many of the models of the present work. These training loss histories can function as a surrogate for the convergence of the ML model with lower training losses corresponding to a closer match between the model reconstructions and the reference DES data for the training data sets of Table 2. Training loss history data is split across three different plots to ensure the legibility of data across all cases and to compartment data for ease of comparison. Figs. 9 and 11 compare the training loss history for the four models, one at each patch size, against the baseline model of [1] and the body-masked model of this paper. Fig. 9 shows losses for the pressure data, and Fig. 11 shows losses for the density data. The model in [1] and the masked models consider full data sets (i.e., rather than using patch-decomposed training data), so these two models serve as baselines against which to evaluate the patch-trained models. For this reason, the figures denote training loss histories from these two models with dotted lines, and they denote the histories from the patch-trained models with solid lines. The three patch-trained models start with the same training loss and they reach lower levels of loss by the end of training in relation to the accuracy of their predictions compared against the DES data. The training losses for the geometry mask and the models of [1] start at different values due to differences in loss function and the inclusion of the second input layer in the geometry mask model.

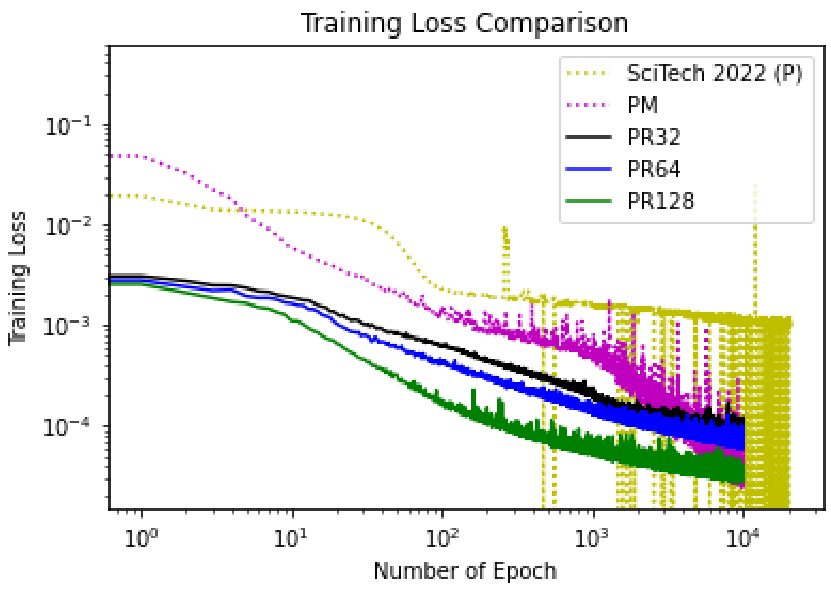

Fig. 9 Training Loss Comparison for *Random* Patch Technique and Pressure

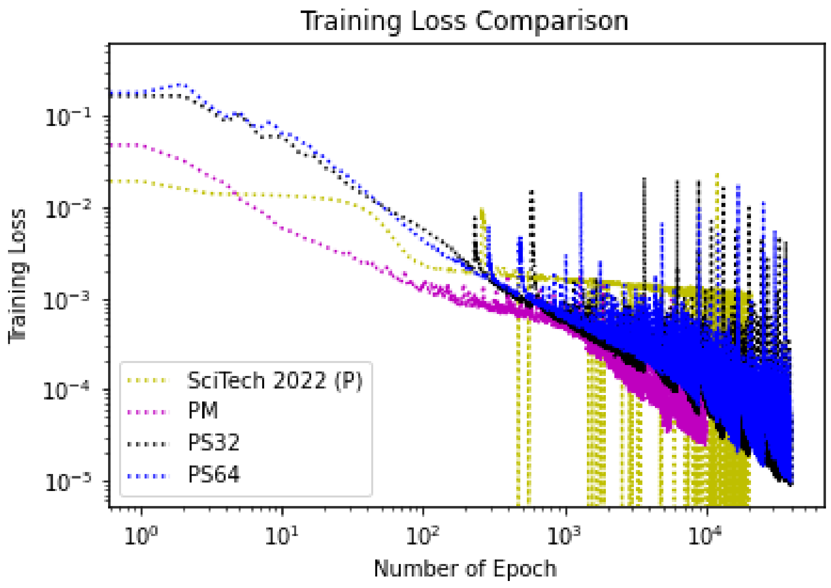

Fig. 10 Training Loss Comparison for *Systematic* Patch Technique and Pressure

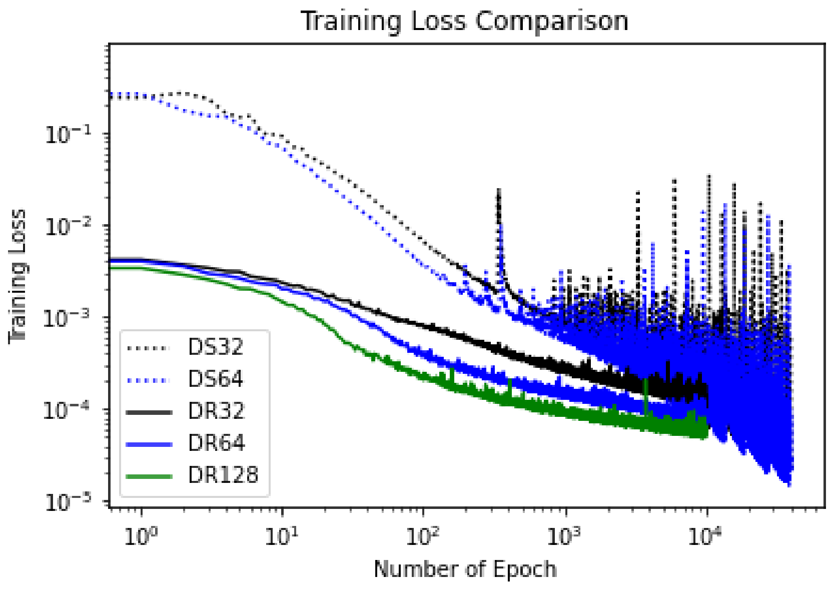

Fig. 11 Training Loss Comparison for *Random* Patch Technique and Density

Figs. 10 and 11 show the training histories for the systematic patched data for pressure and density, respectively. The loss histories start at higher values with the systematic patches compared to the random patches because fewer data points are available within each training epoch. This means that the random patch data set results in more weight and bias updates during each epoch than the systematic patch data set. Both the `32x32` and `64x64` patch sizes resulted in similar training loss histories. No correlation occurs between the degree of training loss reduction and the predictive performance of the final trained model as is the case for the random patched data. The systematic data training loss histories do, however, tend to resolve to lower values than the random data histories. This tracks well with the qualitative results of Figs. 4 and 7, where both the pressure and density predictions from the models trained with systematic patched data perform better than those trained with random patched data.

V. Conclusions

This study presented several CNN models that aimed at post-processing URANS CFD results to generate higher fidelity flow fields with reduced computational cost compared to running DES simulations. The study investigated several pre-processing steps to improve the predictive capability of previously presented models [1]. Examining pressure and density fields, rather than vorticity magnitude fields considered but [1], provided more intuitive points of comparison among URANS, DES, and CNN prediction results, and it provided predicted data useful for calculating quantities of interest on the wall region in future studies. Performing hyperparameter optimization using SMAC before fully training the CNN models resulted in meaningful improvements in model predictive capability. In general, the systematic patching pre-processing technique produced better performing models than the random patching pre-processing technique. Including a geometry-masking layer in the model inputs did initially help improve the ability of the trained models to replicate the airfoil geometries accurately, but this ultimately did not prove necessary. An increased number of training epochs along with better refined hyperparameter optimization resulted and paired with patched-training-data-produced trained models that accurately predicted the airfoil geometries considered in this study.

The improvements considered in this study do represent an improvement over the model presented at SciTech 2022 [1]. However, a need for improvement remains. Further study should seize the potential of using CNN models to generate higher resolution CFD predictions. Future work should revolve around the following: (1) continue refining the hyperparameter optimization framework defining CNN model structure, (2) create additional post-processing steps to calculate pressure effects on the airfoil wall, (3) compare these results against validated simulation and experimental force/moment data, and (iv) extend the current modeling paradigm to a greater number of inputs so that a single CNN model can simultaneously handle all primitive variables.

Acknowledgments

The authors thank Dr. Christopher Rumsey and Dr. Gary Coleman of NASA Langley Research Center for their guidance in conducting this research. The authors also thank their colleagues at the Naval Surface Warfare Center, Dahlgren Division, for their insightful discussion and suggestions in adding a geometry-masking layer to the CNN model. Professor Christopher Roy of Virginia Polytechnic Institute and State University provided guidance to A. C. Brodeur during the course of this research as his major advisor. J. P. Romano and A. C. Brodeur gratefully acknowledge the Office of Naval Research whose sponsorship and continuing guidance of the In-House Laboratory Independent Research (ILIR) program has made this research possible, along with the Naval Surface Warfare Center, Dahlgren Division, for their continued support of this ILIR effort. The authors also acknowledge the U.S. Department of Defense's HPCMP and Old Dominion University's High Performance Computing service for providing computational resources.

References

[1] Romano, J., and Baysal, O., "Convolutional-neural-network-based Auto-encoder for Synthetic Upscaling of Computational Fluid Dynamics Simulations," *AIAA SCITECH 2022 Forum*, 2022, p. 0186.

[2] Duraisamy, K., Iaccarino, G., and Xiao, H., "Turbulence Modeling in the Age of Data," 2019, pp. 1–23.

[3] Brunton, S. L., Noack, B. R., and Koumoutsakos, P., "Machine Learning for Fluid Mechanics," *Annual Review of Fluid Mechanics*, Vol. 52, No. 1, 2020, pp. 477–508. https://doi.org/10.1146/annurev-fluid-010719-060214.

[4] Taira, K., Hemati, M. S., Brunton, S. L., Sun, Y., Duraisamy, K., Bagheri, S., Dawson, S. T., and Yeh, C. A., "Modal analysis of fluid flows: Applications and outlook," *AIAA Journal*, Vol. 58, No. 3, 2020, pp. 998–1022. https://doi.org/10.2514/1.J058462.

[5] Erichson, N. B., Mathelin, L., Yao, Z., Brunton, S. L., Mahoney, M. W., and Kutz, J. N., "Shallow neural networks for fluid flow reconstruction with limited sensors," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, Vol. 476, No. 2238, 2020, p. 20200097. https://doi.org/10.1098/rspa.2020.0097.

[6] Singh, A. P., Medida, S., and Duraisamy, K., "Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils," *AIAA Journal*, Vol. 55, No. 7, 2017. https://doi.org/10.2514/1.J055595.

[7] Tracey, B., Duraisamy, K., and Alonso, J. J., "A machine learning strategy to assist turbulence model development," *53rd AIAA Aerospace Sciences Meeting*, , No. January, 2015, pp. 1–23. https://doi.org/10.2514/6.2015-1287.

[8] Duraisamy, K., Zhang, Z. J., and Singh, A. P., "New approaches in turbulence and transition modeling using data-driven techniques," *53rd AIAA Aerospace Sciences Meeting*, , No. January, 2015, pp. 1–14. https://doi.org/10.2514/6.2015-1284.

[9] Zhang, Z. J., and Duraisamy, K., “Machine learning methods for data-driven turbulence modeling,” *22nd AIAA Computational Fluid Dynamics Conference*, , No. June, 2015, pp. 1–18. https://doi.org/10.2514/6.2015-2460.

[10] Tracey, B., Duraisamy, K., and Alonso, J. J., “Application of supervised learning to quantify uncertainties in turbulence and combustion modeling,” *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition 2013*, , No. January, 2013, pp. 1–18. https://doi.org/10.2514/6.2013-259.

[11] Rumsey, C. L., Coleman, G. N., and Wang, L., “In Search of Data-Driven Improvements to RANS Models Applied to Separated Flows,” *AIAA SCITECH 2022 Forum*, 2022, p. 0937.

[12] Pochampalli, R., Oezkaya, E., Zhou, B. Y., Suarez Martinez, G., and Gauger, N. R., “Machine learning enhancement of Spalart-Allmaras Turbulence Model using Convolutional Neural Network,” *AIAA Scitech 2021 Forum*, 2021, pp. 1–17. https://doi.org/10.2514/6.2021-1017.

[13] Agostini, L., “Exploration and prediction of fluid dynamical systems using auto-encoder technology,” *Physics of Fluids*, Vol. 32, No. 6, 2020, p. 067103. https://doi.org/10.1063/5.0012906, URL https://doi.org/10.1063/5.0012906.

[14] Fukami, K., Fukagata, K., and Taira, K., “Super-resolution reconstruction of turbulent flows with machine learning,” *Journal of Fluid Mechanics*, Vol. 870, 2019, pp. 106–120. https://doi.org/10.1017/jfm.2019.238.

[15] Matsuo, M., Nakamura, T., Morimoto, M., Fukami, K., and Fukagata, K., “Supervised convolutional network for three-dimensional fluid data reconstruction from sectional flow fields with adaptive super-resolution assistance,” 2021. URL http://arxiv.org/abs/2103.09020.

[16] Lui, H., and Wolf, W., “Convolutional Neural Networks for the Construction of Surrogate Models of Fluid Flows,” *AIAA Scitech 2021 Forum*, 2021, pp. 1–15. https://doi.org/10.2514/6.2021-1675.

[17] Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., and Hoyer, S., “Machine learning–accelerated computational fluid dynamics,” *Proceedings of the National Academy of Sciences*, Vol. 118, No. 21, 2021, p. e2101784118. https://doi.org/10.1073/pnas.2101784118, URL http://www.pnas.org/lookup/doi/10.1073/pnas.2101784118.

[18] Biedron, R. T., Carlson, J.-R., Derlaga, J. M., Gnoffo, P. A., Hammond, D. P., Jones, W. T., Kleb, B., Lee-rausch, E. M., Nielsen, E. J., Park, M. A., Rumsey, C. L., Thomas, J. L., and Wood, W. A., “FUN3D Manual: 13.1,” Tech. rep., 2017.

[19] Hutter, F., Lücke, J., and Schmidt-Thieme, L., “Beyond Manual Tuning of Hyperparameters,” *KI - Kunstliche Intelligenz*, Vol. 29, No. 4, 2015, pp. 329–337. https://doi.org/10.1007/s13218-015-0381-0.

[20] Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., and Hutter, F., “SMAC3: A Versatile Bayesian Optimization Package for Hyperparameter Optimization,” *ArXiv: 2109.09831*, 2021. URL https://arxiv.org/abs/2109.09831.

[21] Hesamian, M. H., Jia, W., He, X., and Kennedy, P., “Deep learning techniques for medical image segmentation: achievements and challenges,” *Journal of digital imaging*, Vol. 32, No. 4, 2019, pp. 582–596.

[22] Viquerat, J., Larcher, A., El Haber, G., and Hachem, E., “Robust Deep Learning For Emulating Turbulent Viscosities,” , Nov. 2021. URL https://hal.archives-ouvertes.fr/hal-03432657, working paper or preprint.

[23] Peng, X., Li, X., Gong, Z., Zhao, X., and Yao, W., “A deep learning method based on patchwise training for reconstructing temperature field,” , 2022. https://doi.org/10.48550/ARXIV.2201.10860, URL https://arxiv.org/abs/2201.10860.