Application of Convolutional Neural Network to Predict Airfoil Lift Coefficient

Yao Zhang*, WoongJe Sung†, Dimitri Mavris‡

School of Aerospace Engineering, Georgia Institute of Technology, Atlanta, GA, 30332

The adaptability of the convolutional neural network (CNN) technique is probed for aerodynamic meta-modeling task. The primary objective is to develop a suitable architecture for variable flow conditions and object geometry, in addition to identifying a sufficient data preparation process. Multiple CNN structures were trained to learn the lift coefficients of the airfoils with a variety of shapes in multiple flow Mach numbers, Reynolds numbers, and diverse angles of attack. This was conducted to illustrate the concept of the methodology. Multi-layered perceptron (MLP) solutions were also obtained and compared with the CNN results. The newly proposed meta-modeling concept has been found to be comparable with the MLP in learning capability; and more importantly, our CNN model exhibits a competitive prediction accuracy with minimal constraints in geometric representation.

I. Nomenclature

α, AoA	=	angle of attack
ρ	=	raw pixel density
$\bar{\rho}$	=	pixel density
c	=	chord length
C_l	=	sectional lift coefficient
CFD	=	computational fluid dynamics
CNN	=	convolutinal neural network
FC	=	fully-connected
M_∞	=	freestream Mach number
MLP	=	multi-layered perceptron
MSE	=	mean squared error
NACA	=	National Advisory Committee for Aeronautics
ReLU	=	rectified linear unit
tanh	=	tangent sigmoid function
trn	=	dataset for training
vld	=	dataset for validation

II. Introduction

THE rise in computational power has undeniably transformed engineering analysis and design. Recently, the rapid advance in data science and machine learning techniques has opened new doors in understanding unsolved aerodynamics and fluid mechanics problems [1–3]. It also allows for more possibilities in terms of predictive and control capability [4]. The traditional way of solving aerodynamics and fluid mechanics problems is a top-down approach that relies on physics modeling. In order to build a good model, a researcher needs to have a clear understanding of physics and theories. Most of these problems are high-dimensional, multi-scale, and nonlinear problems, which can be very difficult to find underlying physics. Some low-fidelity models are based on a first-order understanding. However, due to the complexity of the real-world problems, there will inevitably be non-negligible differences between the actual dynamics and the approximated models. Thus, the goal of meta-modeling, in this context, is to balance the modeling

---

*Ph.D.Candidate, yzhang401@gatech.edu, Student Member AIAA

†Research Engineer II, woongje.sung@asdl.gatech.edu, Member AIAA

‡Regents Professor for Advanced Systems Analysis, AIAA Fellow

efforts and scope of the design space. This is a difficult task [5] that begs the question of whether there is a way to look at the problem from a different perspective. It is believed that with the help of data science and machine learning, there may be a way.

For the past decade, the potential of combining the meta-modeling of CFD tools with machine learning techniques has gained widespread attention. Data-driven surrogate models are becoming popular, due to the high volume of physical testing and the simulation data generated from the design, analysis and optimization processes. Different machine learning algorithms are being proposed to solve aerodynamics problems. In addition, deep learning techniques have been brought to many researchers' attention [1, 6]. Most of the existing interdisciplinary work, in terms of the combination of meta-modeling of CFD and machine learning techniques, adopts the learning architecture that belong to the category of the 'vanilla' multi-layered perceptron (MLP). MLP is a straightforward, fully-connected neural network architecture that is not specifically designed to exploit spatial and/or temporal correlation that are intrinsic in many real-world problems. For example, Müller et al. applied MLP approaches to construct low-order models describing the near wall dynamics in turbulent flows [3]. Rai and Madavan have shown the potential of using multi-layered neural networks with the 2D aerodynamic design of turbomachinery [7, 8]. Their results demonstrate the advantage of the capability and efficiency to achieve design targets through the MLP architecture. The MLP learning technique does work, but there still is much room for improvement. This improvement may include faster training and reduction in the amount of simulation data required for prediction accuracy in a design space [7]. In the MLP architecture, the learning capability can be increased by additional hidden layers and/or additional hidden units in each hidden layer. However, the trade-off space between the network size and the learning capability is highly quantized with stereotyped variability due to the underlying assumption of 'fully-connected' network structure. In the other hand, the convolutional neural network (CNN) provides an attractive alternative to the MLP and rapidly replacing the standard MLP technique in many challenging machine-learning tasks such as image recognition [9, 10]. In this investigation, we propose a methodology applying the convolutional neural network (CNN) framework to the meta-modeling of fluid mechanics analysis tools. Previous work [11, 12] has illustrated that the CNN enables to learn invariant high-level features when the data have strong spatial and/or temporal correlations. This ability is crucial for many aerodynamic meta-modeling tasks as much as for other real-world learning tasks. Guo et al. applied CNN to approximate the steady state laminar flow [13]. This work has shown that CNNs enable a rapid estimation for the flow field, and the study focused on the visual replication of the velocity field around a group of categorical geometries. Yilmaz and German [14] applied CNN architecture to map airfoil shapes to pressure distribution under the framework of classification problem using discretized pressure coefficient. This study learned and predicted pressure distribution on airfoils in zero-angle of attack only. These recent work has demonstrated the increasing attention to the CNN techniques in the fluid mechanics domain. This attention partly originates from the potential benefit of the CNN's flexibility in geometric representation and scalability for the larger problems beyond current capabilities. *In this work, multiple CNN structures were trained to predict the lift coefficients of airfoils with a variety of shapes in multiple freestream Mach numbers, Reynolds numbers, and diverse angles of attack.*

III. Methods

A. Convolutional Neural Network Architecture

Since the seminal introduction of the LeNet-5 architecture [9], the CNN has widely been used for challenging machine-learning tasks, especially in image recognition applications. It is one of the core computational structures enabling a significant share of modern deep learning techniques [10]. The CNN can capture the invariance from complex and noisy real-world data in a scalable manner, due to a unique sub-structure of local receptive fields with weight-sharing and subsequent mapping through sub-sampling layers [15].

Fig. 1 illustrates the simplified layout of the LeNet-5 applied to the handwritten digit recognition task [9]. Beyond this structure, aerodynamic meta-modeling requires various modifications. Besides object geometry, flow conditions need to be considered. In addition, the network output needs to be a continuous regression instead of a discrete classification. Fig. 2 is a conceptual CNN architecture reflecting those modifications on top of the LeNet-5 where, in contrast to the conventional aerodynamic meta-modeling, the object geometry is represented by an image-like array of pixels instead of a set of coordinates, shape functions, or pre-defined parameters.

For the actual implementation, three different neural network structures were designed to learn and predict the lift coefficient of the airfoil, as shown in Fig. 3. The 'MLP' represents a conventional multi-layered perceptron, which has two hidden layers between the input and output layers. Here 'FC' means a fully connected layer where each and
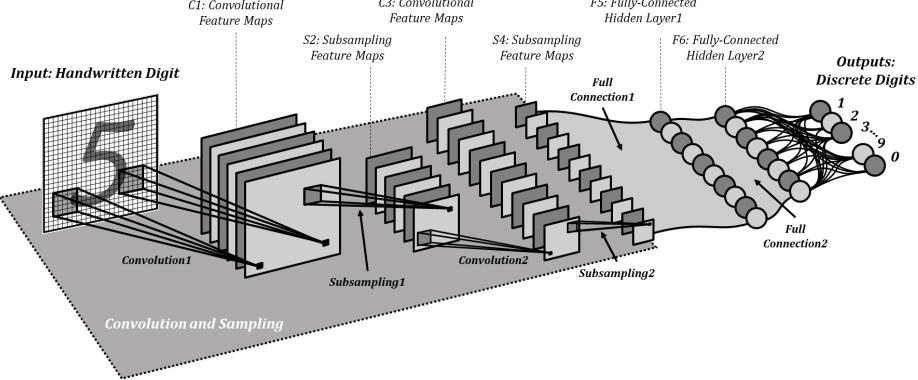
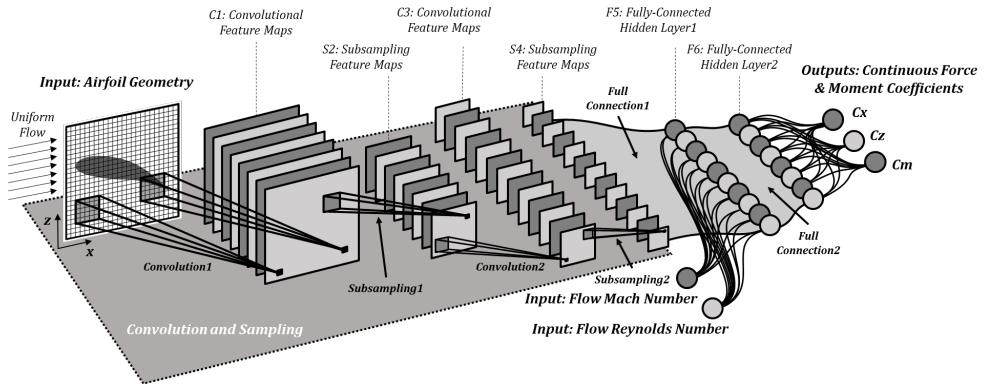**Fig. 1 LeNet-5[9]: CNN for handwritten digit recognition task**

Fig. 2 Modified LeNet-5 for a notional aerodynamic meta-modeling task

every units in one layer connects to each and every units in the other layer. In this work, the *MLP* serves as a baseline architecture to assess the relative performance of the other networks. 'AeroCNN-I' and 'AeroCNN-II' are two types of simple CNN architectures with different convolution schemes and internal layouts. Each architecture is designed to process a unique input data structure, hence, requiring different process for its data preparation. *MLP* uses the freestream Mach number, Reynolds number, angle of attack, and 100 y-coordinates (50 points for upper surface and 50 points for lower surface) around each airfoil at pre-defined 50 x-positions. These 50 positions are mildly clustered toward the leading edge considering the higher variability in airfoil geometry near its leading edge. This results in total 103 input parameters. *AeroCNN-I* has the same 3 parameters for the flow condition and the same 100 y-coordinates. However, the y-coordinates has a form of the 2D array (2×50) structure convoluted by the local 2D kernels having the dimension of (2×5). Here, the first row corresponds to the lower surface y-coordinates and the second row to the upper surface. *AeroCNN-I* is an intermediate architecture between the conventional MLP and fully-convoluting *AeroCNN-II*. Fig. 4 illustrates a typical convolution scheme used in image recognition tasks where the local receptive field with a kernel of (3×4) maps to 3 filters that are sub-sampled by (2×2) pooling. This 'pooling' operation is usually executed, in order to reduce dimensionality of the learning process, by averaging or taking the maximum value. For *AeroCNN-II* only, an 'artificial image' for each data entry is pre-processed to (49×49) pixels of 2D array. (5×5) kernels, 25 filters, and (2×2) 'max' pooling have been chosen after a series of numerical experiments to ensure a reasonable prediction performance. The number of 'convolution-pooling' pairs has also been chosen as one by the same reason, i.e., two pairs of 'convolution-pooling' that is typical in many image recognition tasks doesn't provide a significant benefit in this case.

AeroCNN-II uses a full image-like 2D array combining the airfoil shape and the flow conditions. For *AeroCNN-II*, to automate the data preparation process and infuse the flow conditions in unified manner, instead of collecting or rasterizing an 'image' of an airfoil, a unique concept of 'artificial image' has been devised and applied to each data entry. The airfoil shape is tilted for the corresponding angle of attack, after which the airfoil shape is converted to an image-like 2D array having a finite number of rows and columns, (49×49) in this work. Finally, the external space is

'colored' by following equation depending on the freestream Mach number and the pixel density;

$$\bar{\rho} = \left(1 - \frac{\rho}{\rho_{max}}\right) \frac{M_\infty}{M_{\infty,max}} \tag{1}$$

where the raw pixel density, ρ, for each cell has the range $[0, 100]$; 100 for the cell located 'completely' inside an airfoil, 0 for the cell located 'completely' outside an airfoil, and $(0, 100)$ for the cell located on the boundary of an airfoil shape. By this mapping, for each airfoil shape, the final pixel density, $\bar{\rho}$, has the common minimum value of 0 for the cell located 'completely' inside the airfoil, but the higher maximum value for the higher freestream Mach number. The overall process is illustrated in Fig. 5 where the external points have different 'color' depending on the corresponding freestream Mach numbers. In this way, the angle of attack, the freestream Mach number, and the airfoil shape are all combined into single data format of the 'artificial image.' For *AeroCNN-II*, the Reynolds number is excluded and only one subset of the entire dataset has been used, i.e., for a single Reynolds number.

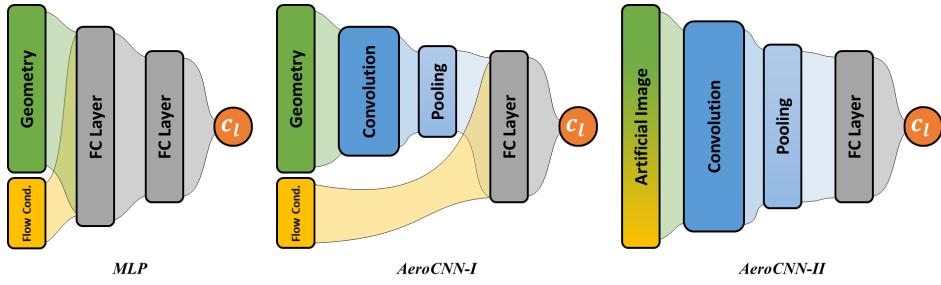All three architectures have been implemented in the Python environment using a modern, open-source library for deep learning: MXNET (http://mxnet.incubator.apache.org).

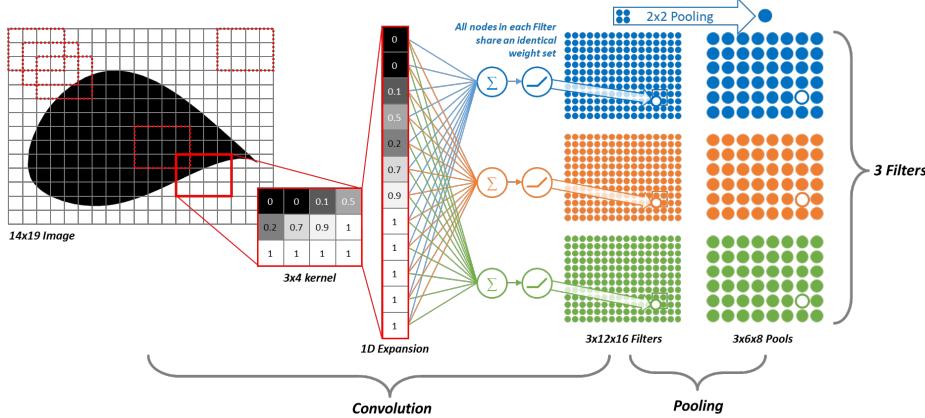**Fig. 3   Three network structures tested in this work**

Fig. 4   Illustration of convolution scheme (Numbers are not actual, but only for illustration purpose.)

B. Data Preparation

The overall data preparation procedures are illustrated in Fig. 6. Obtaining an accurate airfoil shape is critical for the network training results. There are different ways to represent the airfoil shape. The most neutral and common way is to represent it as coordinate points along the upper and the lower surfaces. In this work, the UIUC Airfoil Data Site [16] serves as the dataset necessary to obtain the airfoil shapes. The database has a collection of point-by-point airfoil coordinates format, with (x, y) coordinates. A total 133 sets of 2D airfoil geometry are used as training and validation datasets. This includes a symmetric airfoil, NACA 4-digit and 6-digit series, and others. All airfoils have outer model line shapes without any high lifting apparatus.
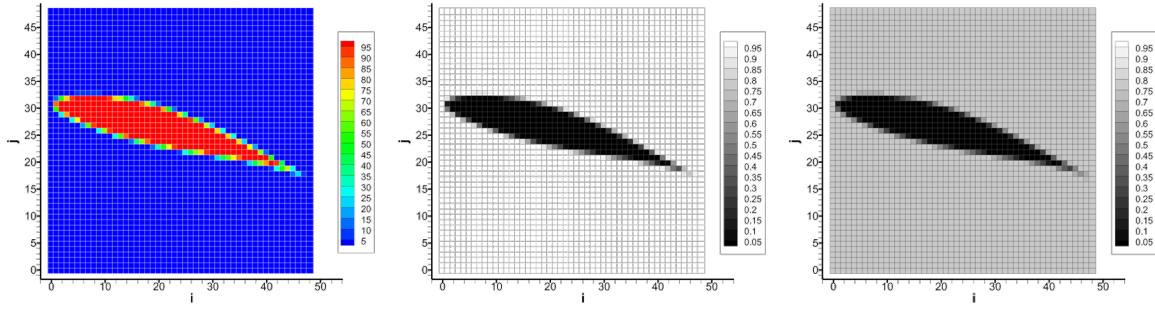
Fig. 5 'Artificial image' of NACA654421a05 at 24°-AoA; (left) tilt by AoA and digitization by pixel density, (middle) nondimensionalized for Mach 0.8, and (right) the same geometry for Mach 0.6

In this work, the aerodynamic coefficient of interest is focused on the lift force coefficient, C_l, to prove the concept of the technique. The collection of the aerodynamic data have been generated from a series of numerical simulations. Due to the nature of this work, the absolute accuracy of the simulation results is not the first priority. Considering the required diversity in flow conditions and airfoil shapes, XFOIL [17] was chosen as the simulation tool to obtain the aerodynamic coefficients. The lift coefficients were obtained for diverse flow conditions for each airfoil using XFOIL as a function of angle of attack ($-10^o - 30^o$), Reynolds number (30,000 - 6,500,000), and freestream Mach number ($0.3 - 0.8$). By running XFOIL and eliminating diverging or non-converging cases, the total number of data entries is about 40,000. To augment the data quality as well as quantity, for each airfoil with a given combination of Mach and Reynolds number, its 'upside-down' or flipped airfoil data has also been created using symmetry in the lift force;

$$C_{l,\text{upside-down}}(\alpha) = -C_{l,\text{original}}(-\alpha) \tag{2}$$

This operation makes the size of entire dataset increase to about 80,000.

C. Activation Functions

In a traditional MLP approach, the hyperbolic tangent (tanh) prevails as the most common choice for the activation function of hidden units. Elman [18] pointed out that the functional shape of the sigmoid-type activators is blamable for the later stagnation in the training process due to its vanishing (or infinitesimal) gradients at the two extrema. Krizhevsky et al. [19] showed the benefit of replacing *tanh* units by the rectified linear unit (ReLU) in image recognition tasks. In the present work, a variety of network structure was tested with both *tanh* and *ReLU* units as the main hidden unit. One training history of the MLP network is shown as a typical example in Fig. 7. Here, the MLP network has 103 input units corresponding to the parameters for airfoil geometry and flow conditions, two hidden layers with 200 and 100 hidden units, and a single output unit of the lift coefficient. It can be seen that the learning capability is significantly higher with the *ReLU* unit although there is an oscillatory behavior. The almost identical benefit with Krizhevsky et al. [19] is obvious for the current task of learning lift coefficients. Therefore, all *MLP*, *AeroCNN-I* and *AeroCNN-II* networks adopt the rectified linear unit as the only activation function except for the pooling and linear units.

IV. Results and Discussion

Both for the MLP and CNNs, weight and biases in the neural networks were trained by *AdaDelta* algorithm [20] that is built in MXNET library, which is a popular choice for modern neural network training due to its flexibility and robustness. It is known that *AdaDelta* algorithm has shown superb error-reduction capability with notable insensitivity to the choices in the hyper-parameters for weight optimization [20]. Under MLP architecture, various structures with different numbers of hidden units have been tested to obtain the overall boundary of the training performance. Fig. 8a presents the training history of those models with a converged mean squared error (MSE) for the training (85%) and validation (15%) datasets. Regardless of the diversity in the number of adjustable parameters or learning capabilities, they show similar learning trajectories resulting in the overall boundary of the obtainable accuracy for the lift coefficient prediction.
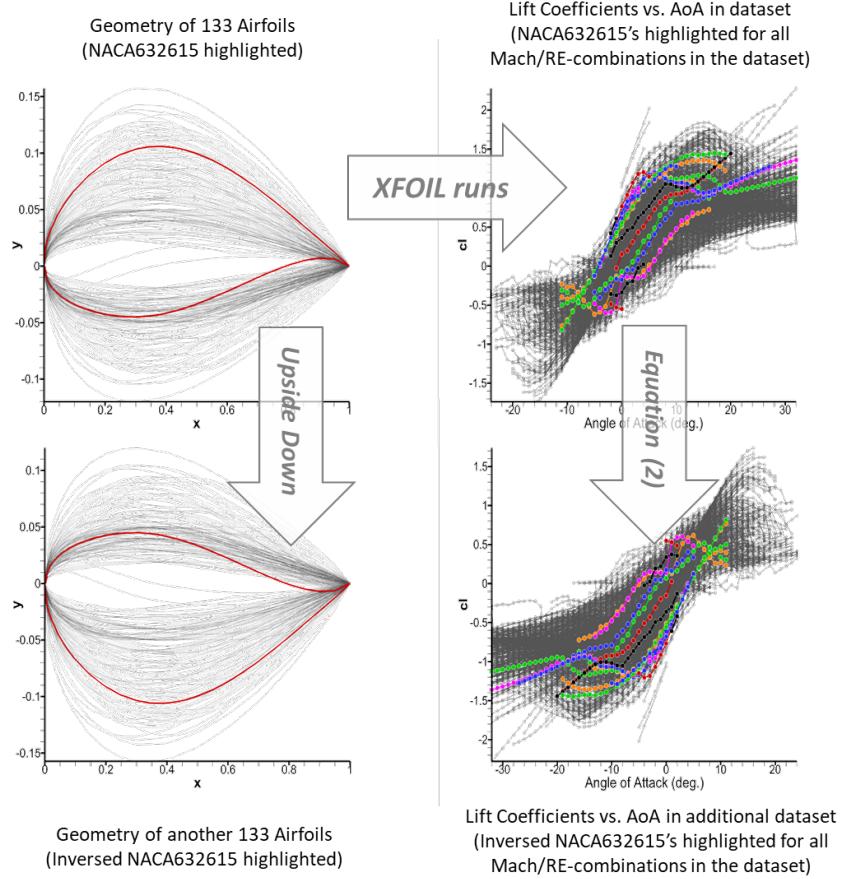
Geometry of 133 Airfoils
(NACA632615 highlighted)

Lift Coefficients vs. AoA in dataset
(NACA632615's highlighted for all
Mach/RE-combinations in the dataset)

XFOIL runs

Upside Down

Equation (2)

Geometry of another 133 Airfoils
(Inversed NACA632615 highlighted)

Lift Coefficients vs. AoA in additional dataset
(Inversed NACA632615's highlighted for all
Mach/RE-combinations in the dataset)

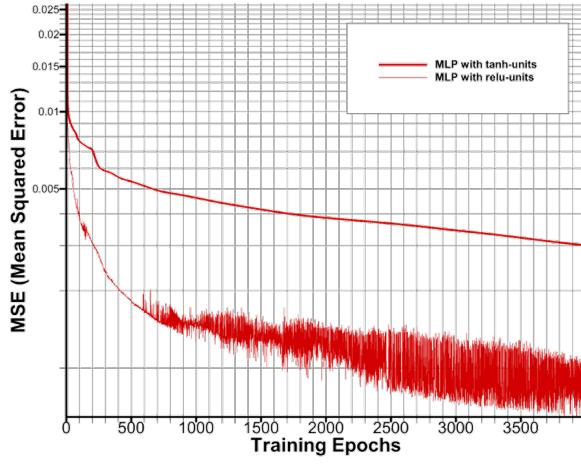**Fig. 6   Data preparation process**

Fig. 7   Typical *MLP* training history with *tanh* and *ReLU* activation

Fig. 8b compares the learning capability between *MLP* and *AeroCNN-I* architectures. The two models have approximately same number of adjustable parameters. In this case, the 30% and 70% split of the entire dataset was used for the training and validation datasets, respectively. It can be seen that, when comparing *MLP* with *AeroCNN-I*, *AeroCNN-I* curves drop faster for both training and validation datasets, before reaching steady MSEs. This implies that

6

AeroCNN-I learns faster than *MLP* for the given number of training epochs. Here, an 'epoch' means one full exposure of whole training examples to the learning architecture. However, the drawback of *AeroCNN-I* architecture is the time consumption. In this case, the total time spent for 500 epochs was 3,206 seconds for *AeroCNN-I* while only 1,700 seconds were necessary for *MLP*. The CNN architecture requires significantly more calculation steps and time per epoch than the MLP architecture for the similar number of adjustable parameters.

The training and validation results from *AeroCNN-II* architecture are plotted in Fig. 9a. *AeroCNN-II* uses full 2D-convolution for the pre-processed 'artificial images.' From Fig. 9a, the performance from *AeroCNN-II* shows a similar training trajectories and comparable results as does *AeroCNN-I* in Fig. 8b. This result suggests that the proposed architecture of *AeroCNN-II* by incorporating the flow conditions and airfoil geometry into the 'artificial image' is a working concept.

A reduced subset of entire dataset for a single Reynolds number was processed through *AeroCNN-II* with 95% of the data used for training and 5% used for validation. The results of the predicted lift coefficients were plotted against the actual lift coefficients obtained from the XFOIL runs. As shown in Fig. 9b, the majority of points are clustered near the 45°-line, meaning that the predicted values are close to the actual values. The training samples indicated by red color fit better than the validation samples indicated by green color. This is a common result in many adaptive machine learning techniques; the generalization for unseen examples is more difficult than just reducing training error. In order to test the prediction capability of *AeroCNN-II*, the lift coefficients are plotted as a function of α for both the CFD results and the prediction from *AeroCNN-II*. Fig. 10 is the $c_l - \alpha$ curve for an airfoil that is included in the training dataset. Fig. 11a and Fig. 11b are for unseen airfoils during the training process. These plots exhibit reasonable training and generalization performance of *AeroCNN-II* architecture for the given task of predicting lift coefficients.

In summary, as a proof of concept, the proposed CNN architectures demonstrates reasonable performance in learning from the given examples and in predicting lift coefficients for unseen airfoil shapes.
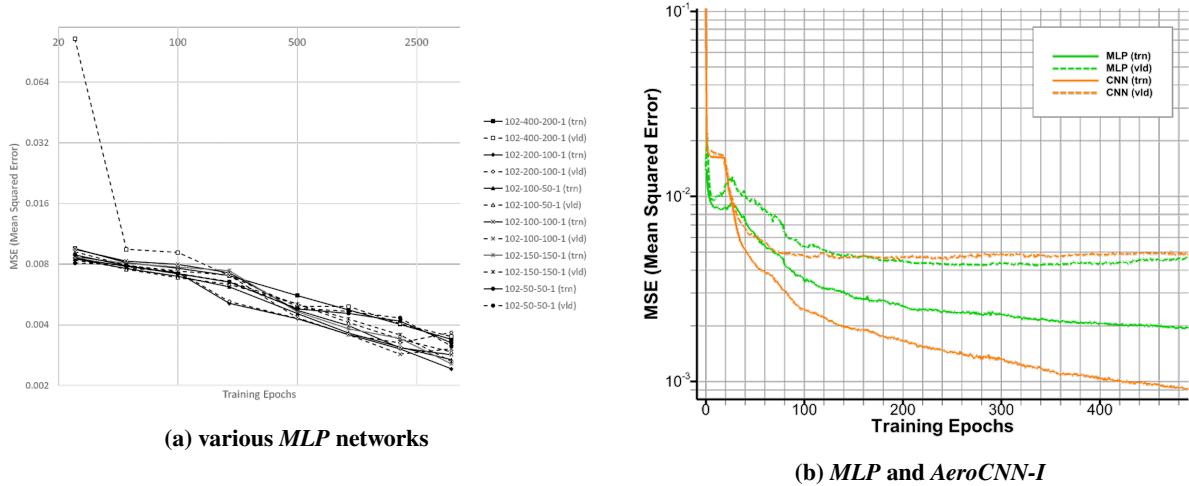

(a) various *MLP* networks

(b) *MLP* and *AeroCNN-I*

Fig. 8   Training history of *MLP* and *AeroCNN-I*

V. Conclusion

Various CNN architecture were applied to study aerodynamics. In this work, they were being applied to predict the lift coefficient with a given airfoil shape and flow conditions. Predictions from both the conventional MLP and the proposed CNN techniques were generated and compared. Two types of CNN architectures were constructed with different convolution schemes and internal layouts. *AeroCNN-II* is the first of its kind to investigate the 2D aerodynamic problems involving diverse flow conditions and the variety of sectional shapes in a single framework of the CNN using the concept of the 'artificial image.'

The synthesis between geometric boundary conditions (e.g., airfoil shape) and non-geometric boundary conditions (e.g., α and M_∞) into an image-like array has been successfully used in training the CNN architecture for aerodynamic meta-modeling task. This result gives a useful perspective to harness well-developed deep learning techniques in image recognition tasks for engineering meta-modeling tasks.
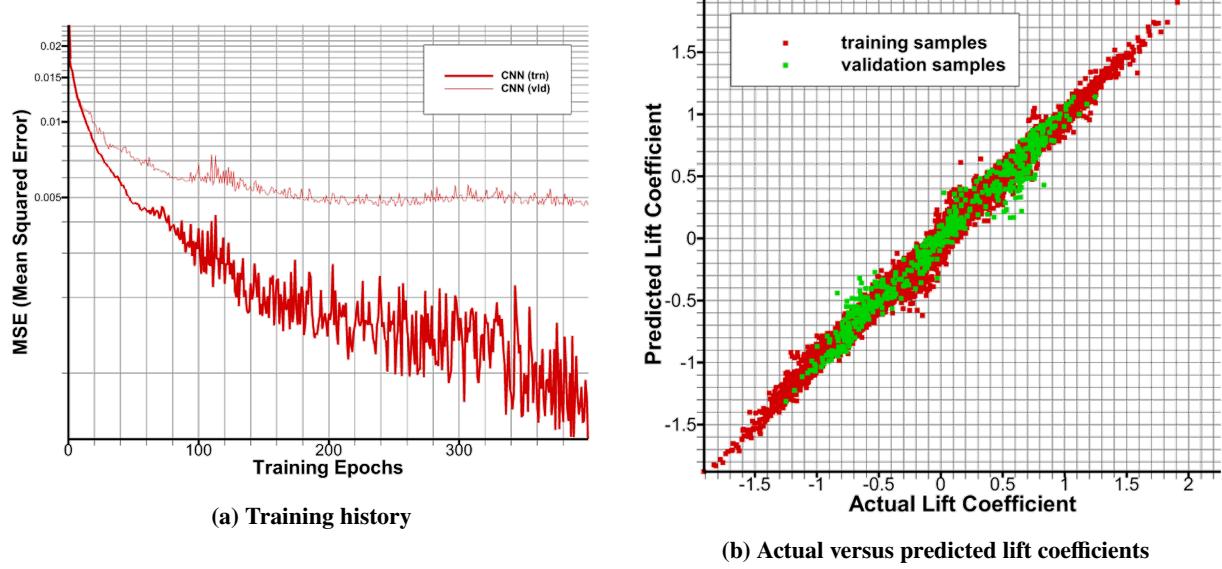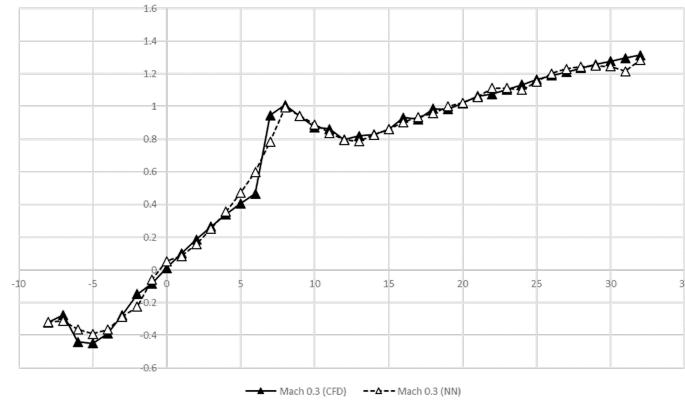
(a) Training history

(b) Actual versus predicted lift coefficients

Fig. 9   Results of *AeroCNN-II* training

Fig. 10   Actual versus predicted $C_l - \alpha$ curve at $M_\infty = 0.3$ for NACA64a410 (included in training dataset)
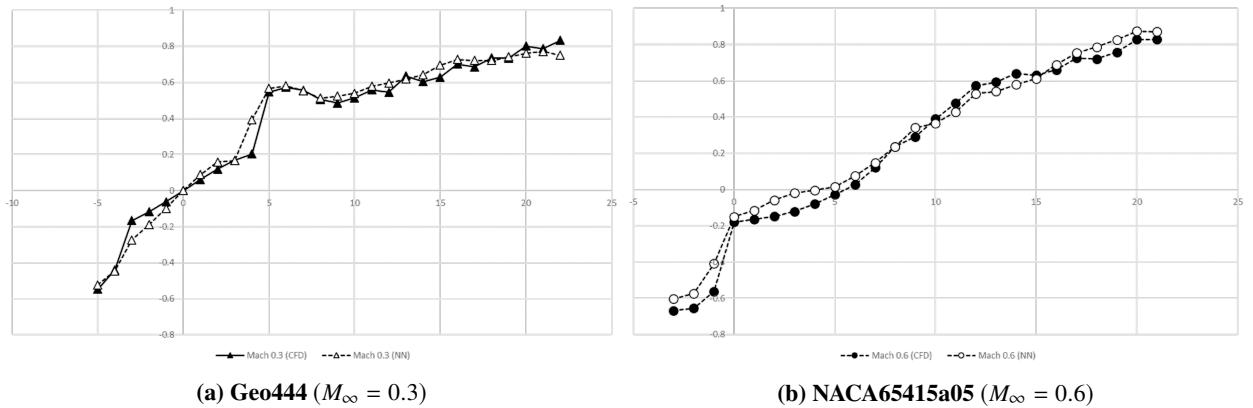

(a) Geo444 ($M_\infty = 0.3$)

(b) NACA65415a05 ($M_\infty = 0.6$)

Fig. 11   Actual versus predicted $C_l - \alpha$ curve (not included in training dataset)

8

References

[1] Zhang, Z. J., and Duraisamy, K., "Machine learning methods for data-driven turbulence modeling," *AIAA Aviation*, Vol. 2460, 2015.

[2] Parish, E. J., and Duraisamy, K., "A paradigm for data-driven predictive modeling using field inversion and machine learning," *Journal of Computational Physics*, Vol. 305, 2016, pp. 758–774.

[3] Müller, S., Milano, M., and Koumoutsakos, P., "Application of machine learning algorithms to flow modeling and optimization," *Center for Turbulence Research Annual Research Briefs, Stanford University, Stanford, CA*, 1999, pp. 169–178.

[4] Gautier, N., Aider, J.-L., Duriez, T., Noack, B., Segond, M., and Abel, M., "Closed-loop separation control using machine learning," *Journal of Fluid Mechanics*, Vol. 770, 2015, pp. 442–457.

[5] Sung, W. J., *A neural network construction method for surrogate modeling of physics-based analysis*, Georgia Institute of Technology, 2012.

[6] Tracey, B., Duraisamy, K., and Alonso, J., "Application of supervised learning to quantify uncertainties in turbulence and combustion modeling," *AIAA Paper*, Vol. 259, 2013, p. 2013.

[7] Rai, M. M., and Madavan, N. K., "Application of artificial neural networks to the design of turbomachinery airfoils," *Journal of Propulsion and Power*, Vol. 17, No. 1, 2001, pp. 176–183.

[8] Rai, M. M., and Madavan, N. K., "Aerodynamic design using neural networks," *AIAA journal*, Vol. 38, No. 1, 2000, pp. 173–188.

[9] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, Vol. 86, No. 11, 1998, pp. 2278–2324.

[10] LeCun, Y., Bengio, Y., and Hinton, G., "Deep learning," *Nature*, Vol. 521, No. 7553, 2015, pp. 436–444.

[11] Zuo, Z., Shuai, B., Wang, G., Liu, X., Wang, X., Wang, B., and Chen, Y., "Convolutional recurrent neural networks: Learning spatial dependencies for image representation," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 18–26.

[12] Taylor, G. W., Fergus, R., LeCun, Y., and Bregler, C., "Convolutional learning of spatio-temporal features," *European conference on computer vision*, Springer, 2010, pp. 140–153.

[13] Guo, X., Li, W., and Iorio, F., "Convolutional neural networks for steady flow approximation," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM, 2016, pp. 481–490.

[14] Yilmaz, E., and German, B., "A Convolutional Neural Network Approach to Training Predictors for Airfoil Performance," *18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2017, p. 3660.

[15] Haykin, S. S., *Neural networks: a comprehensive foundation*, Tsinghua University Press, 2001.

[16] Selig, M. S., *UIUC Airfoil Data Site*, Department of Aeronautical and Astronautical Engineering University of Illinois at Urbana-Champaign, 1996.

[17] Drela, M., "XFOIL: An analysis and design system for low Reynolds number airfoils," *Low Reynolds number aerodynamics*, Springer, 1989, pp. 1–12.

[18] Elman, J. L., "Learning and development in neural networks: The importance of starting small," *Cognition*, Vol. 48, No. 1, 1993, pp. 71–99.

[19] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[20] Zeiler, M. D., "ADADELTA: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.