Aerospace Science and Technology 105 (2020) 105949

Contents lists available at ScienceDirect

Aerospace Science and Technology

www.elsevier.com/locate/aescte

Fast pressure distribution prediction of airfoils using deep learning

Xinyu Hui [a], Junqiang Bai [a,*], Hui Wang [a], Yang Zhang [b,c]

[a] School of Aeronautics, Northwestern Polytechnical University, 127 Youyixi Road, Xi'an, 710072, China
[b] State Key Laboratory for Strength and Vibration of Mechanical Structures, School of Aerospace Engineering, Xi'an Jiaotong University, Xi'an, 710049, China
[c] National Key Laboratory of Science and Technology on Aerodynamic Design and Research, Xi'an, 710072, China

ARTICLE INFO

Article history:
Received 10 December 2019
Received in revised form 17 May 2020
Accepted 22 May 2020
Available online 2 July 2020
Communicated by Qiulin Qu

Keywords:
Machine learning
Convolutional neural network
Aerodynamic design
Pressure distribution prediction

ABSTRACT

In the aerodynamic design, optimization of the pressure distribution of airfoils is crucial for the aerodynamic components. Conventionally, the pressure distribution is solved by computational fluid dynamics, which is a time-consuming task. Surrogate modeling can leverage such expense to some extent, but it needs careful shape parameterization schemes for airfoils. As an alternative, deep learning approximates inputs-outputs mapping without solving the efficiency-expensive physical equations and avoids the limitations of particular parameterization methods. Therefore, this paper presents a data-driven approach for predicting the pressure distribution over airfoils based on Convolutional Neural Network (CNN). Given the airfoil geometry, a supervised learning problem is presented for predicting aerodynamic performance. Furthermore, we utilize a universal and flexible parametrization method called Signed Distance Function to improve the performances of CNN. Given the unseen airfoils from the validation dataset to the trained model, our model achieves predicting the pressure coefficient in seconds, with a less than 2% mean square error.

1. Introduction

In the aerodynamic design optimization, specifying airfoils with well-performance pressure, distributions at the cruise point is an important aspect. A successful target pressure distribution renders favorable reasonable geometry and aerodynamic characteristics. Generally, computational fluid dynamics (CFD) solver handles the computation of aerodynamic forces by solving Navier-Stokes (NS) equations. However, CFD simulation is relatively time-consuming. For example, an inverse problem requiring possibly thousands or more simulations, the overall computation time would be orders of magnitude higher than real-time requirements [1]. Therefore, there is a demand for a fast and accurate pressure predicting method.

With the rise of machine learning (ML) theory and computer science, there has been a rapid development in the field of data processing. The techniques of machine learning and the increase of numerical simulation data amount naturally lead to the applications of data-driven modeling in physical systems, including the aerodynamic design. Unlike CFD methods who strictly follow the physical laws, ML can blindly do pure input-output (I/O) mapping, without incorporating any priori knowledge. Once the model learns the aerodynamic features, it holds the fast prediction ability, which satisfies the unseen data from a validation dataset.

So far, there have been many applications of machine learning in the field of fluid mechanics. Researchers have performed machine learning techniques to improve the accuracy of Reynolds-averaged Navier–Stokes (RANS) [2] models. In the era of big data, data-driven methods tell many successful stories while leaving many challenges. The Gaussian process (GP) [3] regression, also known as Kriging in geostatistics [4], and the surrogate model in the fluid mechanics and computer experiments [5] field, is a non-parametric statistical model which has been extensively used in various cases. For example, Chiplunkar et al. [6] utilized it to predict prediction in fast fluid structure interaction simulations. Liu et al. [7] constructed multiple response surfaces for airfoil design with a multiple-output Gaussian-process-regression model to predict lift, drag, and pitching-moment coefficients. Also, GP is widely used in optimization [8], such as Jouhaud et al. [9], Leifsson et al. [10] and Liu et al. [11] optimization under subsonic, transonic, and hypersonic flow using GP. Tao et al. [12] proposed a multi-fidelity surrogate-based optimization framework for the robust optimizations for airfoil and wing under uncertainty of Mach number.

In this paper, neural networks are chosen to be the core of our method. The neural network has drawn much attention by its accuracy and efficiency. For example, Ling et al. [13] might be the first to apply the deep learning technique to quantify the uncertainties and improve the performances in RANS turbulence

* Corresponding author.
E-mail addresses: xinyu.hui@mail.nwpu.edu.cn (X. Hui), juanqiang@nwpu.edu.cn (J. Bai), wanghui2018@nwpu.edu.cn (H. Wang), youngz@xjtu.edu.cn (Y. Zhang).

https://doi.org/10.1016/j.ast.2020.105949
1270-9638/

modeling. Wu et al. [14] tried to augment turbulence models with physics-informed machine learning. Cruz et al. [15] chose to reconstruct Reynolds force vector by a neural network, to help to improve the RANS fidelity. Besides, researchers have applied many data-driven machine learning methods for design and prediction. For example, Rai et al. [16] used Multilayer Perceptron (MLP) [17] and efficiently achieved the design targets in the aerodynamic design problem. Miyanawala et al. [18] presented a machine learning technique for model reduction of RANS functions for unsteady flow. The results demonstrated the predictive capability of the deep learning technique. Sun et al. [19] introduced an inverse design of airfoils/wings using neural networks. This method performed higher accuracy and better efficiency than many conventional methods. Yan et al. [20] emphasized the use of machine learning techniques in aerodynamic shape optimization, and this approach decreased the requirement of CFD calls and the time cost. Zhou et al. [1] presented an ice detection system via Bayesian neural networks, which could offer both the prediction results and uncertainties rapidly, closed the gap to real-time requirements. Pesce et al. [21] presented a technique to estimate the system states combining the neural networks. The proposed solution was capable of reconstructing the dynamics of a spacecraft with enough accuracy. Yu et al. [22] used deep neural networks to give an accurate prediction of flight trajectories while reducing computational cost. Mannarino et al. [23] developed a neural network-based aerodynamic reduced order model, and this method is rather robust. Habibnia et al. [24] conducted an artificial neural network to interpret the operating conditions in the ground effect.

Kharal et al. [25] also studied an inverse design problem. Sekar et al. [26] explored the ability of the deep learning for the flow field prediction of varying airfoil geometries for the first time. Guo et al. [27] utilized the convolutional neural network to predict the velocity flow field over various geometries, given a good example of employing CNN's ability to learn geometrical details. Sekar et al. [28] used CNN to do the inverse design of airfoils, studied the effects of how the parameter setting affects prediction accuracy. In fact, the research work that adopts data-driven method to improve or replace RANS model, and quantify model uncertainties, is mainly carried out in the past five years [29] [30]. Ma et al. [31] explored a closure model for two-fluid multiphase flow equations for a bubbly system. Maulik et al. [32] utilized neural networks to perform convolution and deconvolution operations in large eddy simulations (LESs). Hennigh et al. [33] used DNNs to compress both the computational time and memory of the Lattice Boltzmann simulations. Zhu et al. [34] accessed the ability of ML method to replace RANS models for high Reynolds number flow conditions. Tompson et al. [35] used the CNN method for calculating the inviscid Euler flow, showing a stable divergence-free velocity field with better accuracy and runtime than the Jacobi method. Jin et al. [36] proposed a $C_p - u$ prediction model using CNN to predict the unsteady flow around the circular cylinder. Bhatnagar et al. [37] used CNN to predict the flow fields around airfoils under RANS flow conditions. These works draw the increasing attention of combining ML techniques in design and fluid areas, especially CNN. Compared to the fully-connected neural networks, CNN demands fewer trainable parameters while providing flexibility in learning complex geometrical profiles. Also, CNN has been proven that it can learn invariant high-level features when the data has strong spatial correlations [38], which draws the attention to apply machine learning techniques in aerodynamic design and fluid dynamics.

In addition, several attempts are made to predict the aerodynamic coefficients with deep learning. Yilmaz et al. [39] built the relationship between aerodynamic features and airfoil with CNN and achieved a testing accuracy of about 80%. They performed the image-to-image regression by using convolution and pooling operations. The input image was represented by the 2-D Cartesian coordinate points, which contained little information about each airfoil's geometrical features. This could constrain the accuracy of test results. Then, the input image was mapped to the output using CNN. The output was the pressure coefficient at different positions. In their case, the test output points were sparsely located on the surfaces, and they could hardly describe a complex pressure distribution with shock waves or other nonlinear features. Zhang et al. [40] proposed three kinds of neural network architectures for predicting the airfoil lift coefficient and had good prediction results. In this method, Cartesian coordinate points and binarization were both tried to represent the airfoil. Their MSE loss of testing was converged to 5×10^{-3}, which was also constrained by the geometrical representation method.

In our current study, we extract the ability of CNN to predict pressure distributions. This approach improves the prediction accuracy of CNN with the Signed Distance Function (SDF) [27]. SDF provides a global representation for different 2D/3D geometry profiles and works efficiently with artificial neural networks. The dataset is simulated under subsonic flow, which contains more nonlinearities than low-speed airfoils problem. Based on our five-layer CNN, the trained network could precisely capture the shock-waves from unseen airfoils who have very different patterns compared to the original airfoil, like double shock waves and the changes of the positions and peaks of shock waves. We obtain a less than 2% mean square error result for the validation test and three orders of magnitudes faster than CFD simulation. Compared to forward neural networks (FNNs), our model obtains more accuracy and efficiency. Our results prove the ability of the convolutional neural networks to learn nonlinear functions.

To further improve the learning quality of CNN, we augment the batch normalization (BatchNorm) in the network. The training process of deep neural networks is a complicated procedure, because the distribution of the network layer's inputs variates, makes it hard to train models with nonlinearities. This phenomenon is referred to as an internal covariate shift (ICS). BatchNorm was introduced by Ioffe [41] in 2015 to fix this problem. This widely adopted technique improves the speed, performance, and stability of neural networks. Through the Batch Normalizing transform, the transformation inserted in the network can represent the identity transform, that is, retain the nonlinearity of the inputs. In 2018, Santurkar [42] tended to re-explain the roots of this fundamental technique. Through a series of experiments, Santurkar demonstrated that BatchNorm brings a critical effect during training: it makes the underlying optimization procedure more stable and smoother by making the gradients more predictive and well-behaved, which enables a more efficient optimization. Regardless of the role of BatchNorm, its success is indisputable.

The rest of this paper is organized as follows: Section 2 gives the theoretical and structural information about our neural network. The precise data processing method is introduced in Section 3. The prediction results and further discussion are presented in Section 4. And the conclusion is given in Section 5.

2. Machine learning method

Generally, machine learning can be broadly divided into two classes: supervised learning and unsupervised learning. Supervised learning is trained with labeled data as input. Unsupervised learning uses unlabeled data, and this technique is known as self-organization and allows modeling probability densities of given inputs. We focus on the supervised learning method, which is trained with paired parametrized airfoils and corresponding pressure distributions. CNN is combined with several specific layers to simulate different algorithms, whose performance is depending on

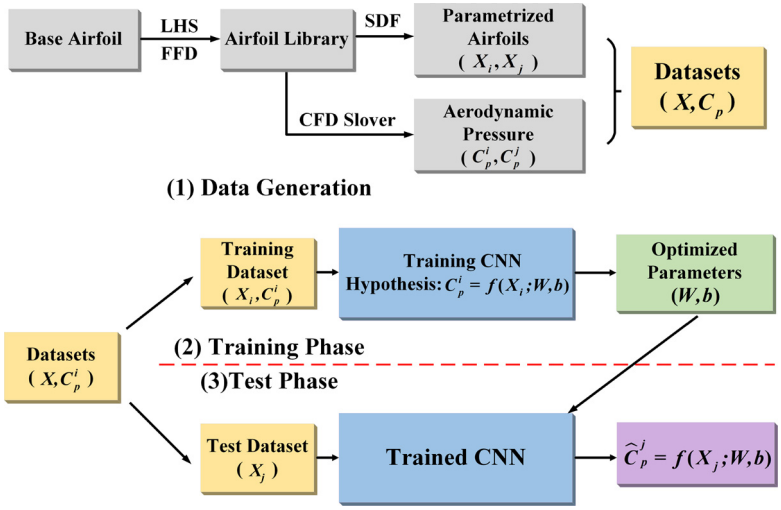


Fig. 1. Training and predicting process.

data and neural network design. The details of our method are given below.

2.1. Pressure distributions prediction model

Our approach uses CNN to simulate the pressure distributions of different airfoil geometries. This method mainly includes three parts: data preparation, training, and validation test. Fig. 1 illustrates the whole procedure of the pressure distribution prediction problem. The learning procedures for a set of samples are summarized as follows:

a) Firstly, the dataset library is formed by deformed airfoils. The deformation method is based on Latin Hypercube Sampling (LHS) [43] and free-form deformation (FFD) [44][45]. Then the airfoils will be parameterized by SDF, these converted airfoils become the input X. In the meantime, the CFD solver will calculate the corresponding aerodynamic feature (C_p). The input map X and the label (C_p) compose the dataset (X, C_p);
b) Secondly, the majority of data is divided into the training dataset, which is marked by the letter i. They will be utilized for training the CNN model under the hypothesis of $C_p = f(X; W, b)$ to optimize the CNN model's parameters;
c) At last, when the training is done, this model can be tested to predict the pressure distribution from unseen airfoils, which are marked by the letter j. The accuracy of the test results will describe the quality of the trained model.

Convolutional neural network mentioned above is the foundation of our method. It is widely used in fields such as image recognition because it can handle image data efficiently. The main idea and operating mechanism of CNN are given below.

2.2. Convolutional neural network

Biological processes inspire the convolutional neural network in the connectivity manner between neurons. These neurons and connections become the organization of the visual cortex [46]. Every neuron only responds to stimuli in a restricted region of the visual field called the receptive field. The receptive fields partially overlap, such that they can cover the entire visual area. Compared to a fully connected network, CNN uses fewer parameters due to its local connectivity and parameter sharing. These features could handle the input information more efficiently. For a given input, CNN will reduce the dimensionalities of the original image, and transform it into a combination of meaningful numbers. The detailed information about the network will be introduced later in this section.

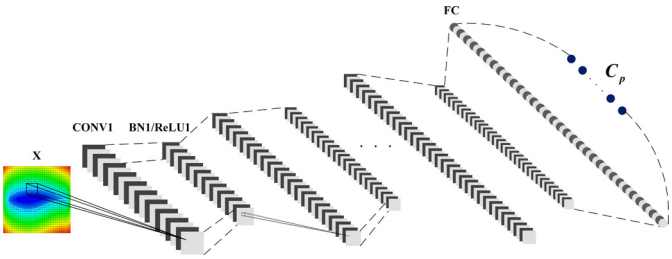


Fig. 2. The CNN architecture used in this paper.

2.2.1. Network structure

To develop suitable CNN architectures for the airfoil shapes, our model is designed to be similar with an encoder. The shown CNN architecture consists of convolutional layers, batch normalization layers, activation layers, a fully connected layer. The sketch of the utilized CNN is shown in Fig. 2.

The inputs to the network are the airfoil shape in the form of an image-like 32×32 2D matrix. Using structured matrixes in CNNs is similar to utilizing images as input, which benefits the encoding of the input into specific properties, and reducing the number of parameters in the network.

Instead of the pooling layer, we choose Batch Normalization to stick with the convolutional layer. Pooled input information likes a summarized version of the features detected in the input. The reason to select Batch Normalization is that the input image only holds the size of 32×32, which is relatively small, the detection of small translations of the input is not necessary, even may affect the accuracy of the model. Further improvements in making this change will be discussed in Section 4.2.

2.2.2. Convolutional layer

The convolutional layer is the core of CNN that does most of the computational lifting. During the convolution process, each filter will slide across the input's width and height, giving us the responses at every spatial position. Through the accumulation of each layer, this network will learn the entire patterns. In order to improve the efficiency of the training, local connectivity and parameter sharing scheme that mentioned before are also included. The local connectivity is used to connect each neuron to a local re-

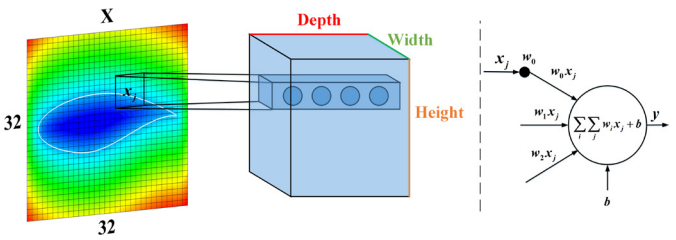


Fig. 3. The left side of the imaginary line: An example of how the neurons work in the first convolutional layer. Each neuron along with the depth only looks at the same part of the region in the input. Right: neurons compute the product of weights and local spatial position. In every depth slice, neurons use the same parameters.

gion in the input. Parameter sharing scheme forces the neurons in every depth slice (denote a single 2-dimensional slice of depth as a depth slice, e.g., a volume of size $[32 \times 32 \times 20]$ has 20 depth slices, each of size $[32 \times 32]$) to share the same parameters, which dramatically reduces the number of parameters. The equation of convolutional operation is given in the following equation, and the working procedure of the first convolutional layer is shown in Fig. 3.

$$y = \sum_i \sum_j w_i x_j + b \tag{1}$$

where y is the output of the convolutional layer, x_j is defined as the local region that each neuron sees, w_i are the weights for different slices, and b is the bias.

2.2.3. Batch normalization and fully connected layers

Batch Normalization is a novel mechanism for accelerating and stabilizing the training process. Over 10,000 citations mentioned BatchNorm provide the evidence of its popularity. Our CNN will use this technique to improve the performance of the prediction ability. The Batch Normalization is defined as Function (2). Through the Batch Normalizing layer, it can be considered that the transformation inserted in the network can represent the identity transform, that is, retain the structures of the nonlinearity of the inputs.

$$\begin{aligned}
\mu &= \frac{1}{m}\sum_{j=1}^{m} x_j && \text{// mini-batch mean} \\
\sigma^2 &= \frac{1}{m}\sum_{j=1}^{m}(x_j - \mu)^2 && \text{// mini-batch variance} \\
\hat{x}_j &= \frac{x_j - \mu}{\sqrt{\sigma^2 + \epsilon}} && \text{// normalize} \\
y_j &= \gamma \hat{x}_j + \beta \rightarrow BN_{\gamma,\beta}(x_j) && \text{// scale and shift}
\end{aligned} \tag{2}$$

where γ and β are learnable parameters which will be optimized during training, in our case, γ can be considered as the weights w_i, and β is the biases b. ϵ is a constant added to the mini-batch variance for numerical stability.

The Rectified Linear Unit (ReLU) [47] activation function is utilized after the BatchNorm layers. The purpose of the activation function is to introduce nonlinearity into the output of a neuron. A two-layer neural network can approximate most of the nonlinear functions when the activation function is nonlinear [48].

Neurons in a fully connected (FC) layer fully connected two adjacent layers, but neurons within a single layer share no connections. In our network, the FC layer is used as the last layer, which compiles the data extracted by previous layers to form the final output.

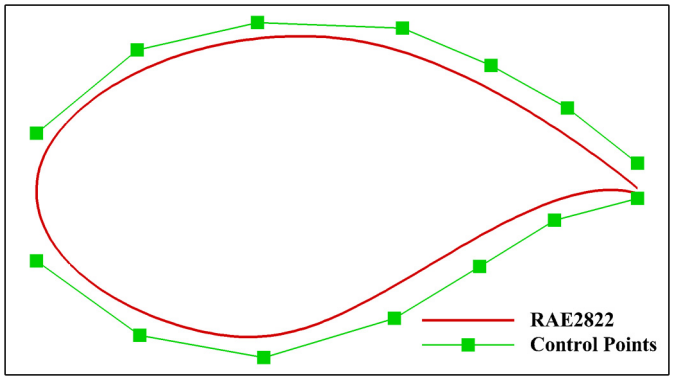


Fig. 4. Control points formed the FFD frame.

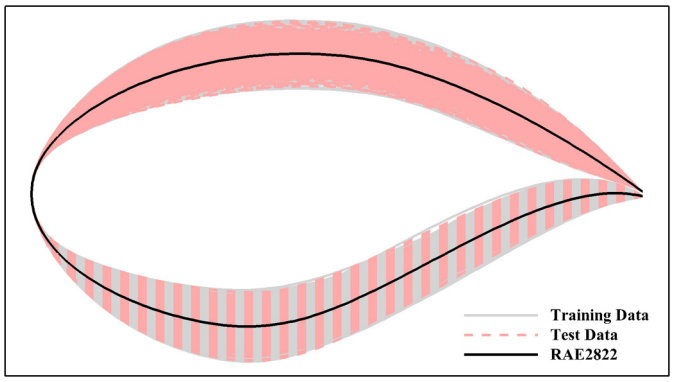


Fig. 5. The airfoil library. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

3. Generation of training and validation samples

Data preparation is an essential aspect of machine learning, which will affect the performance of training. The present approach consists of two stages: the parametrization of the airfoils is demonstrated in the first stage, and aerodynamic simulation is performed in the second stage.

3.1. Airfoil representation stage

Given an original airfoil, the airfoil library is generated by free-form deformation deformed airfoils. FFD is a powerful tool in the geometric design of 2D and 3D objects. The general idea is to deform the shape of an embedded object in a hull by manipulating the control points on the shell. We choose RAE2822 as the base airfoil. Fig. 4 illustrates the control points of the FFD frame. Sixteen control points in the middle of the airfoil (including upper and lower surface, control points near the leading and trailing edge stay fixed) can move perpendicular in the range of $[-0.02, 0.02]$. The moving distances are sampled with Latin Hypercube Sampling. The library is formed by a total number of 1500 airfoils, 2/3 of the library are used for training; the rest are divided into the validation dataset. Fig. 5 shows the airfoil library used in this paper.

CNNs have many successful applications in geometry representation learning and per-pixel prediction in images. Given the coordinate points of airfoils along the upper and lower surfaces, there are multiple ways to represent airfoil geometry. The most common solution is to use some sampling points along the upper and lower surface [39] directly. Zhang [40] presented the binarization method, which filled each cell in a 2D image-like array due to the percentage of the area occupied by the inside of the airfoil. However, these representations are not effective enough for

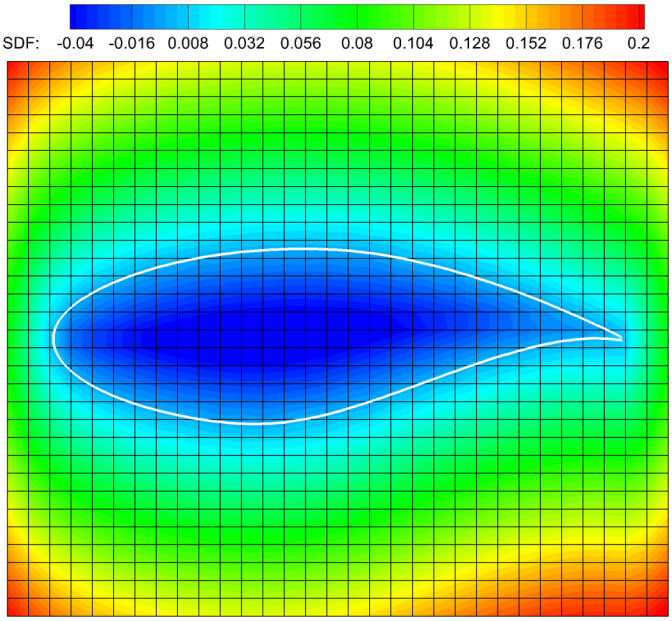


Fig. 6. The SDF representation of base airfoil RAE2822 in a 32×32 Cartesian grid. The RAE2822 boundary is shown in white.

neural networks because they contain blurry and less geometry information. Such as binarization, only the cells around the airfoil boundary were effective; the rest areas share the same one or two values. For example, a 49×49 array may only contain 100 useful messages. Thus, we use Signed Distance Function [27] to represent airfoil geometry. In this method, airfoils are represented by a 32×32 Cartesian grid, which works efficiently with neural networks.

The mechanism of SDF is given below. An airfoil geometry can be considered as a closed area. A set of points (i, j) in the domain Ω describes the airfoil boundary:

$$Z = \{\Pi_\Omega(i, j) = 0 | (i, j) \in R^2, \Omega \in R^2\} \tag{3}$$

where Π_Ω is defined as the airfoil boundary function. $\Pi < 0$ if and only if (i, j) is inside the geometry, $\Pi > 0$ if and only if (i, j) is outside the geometry.

A Signed Distance Function $D(i, j)$ associates to the airfoil boundary function Π_Ω is defined by Equation (4).

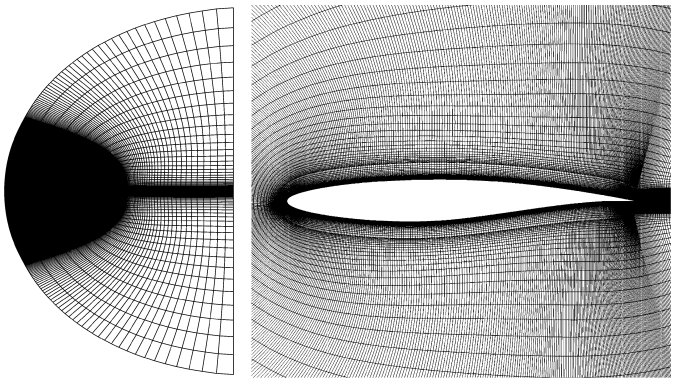
$$D(i, j) = \min_{(i', j') \in \Omega} \| (i, j) - (i', j') \| \, sign(\Pi(i, j)) \tag{4}$$

$D(i, j)$ measures the distance from (i, j) to its nearest point (i', j') on the boundary. *sign* determines whether the point is inside or outside the airfoil. The SDF representation of base airfoil RAE2822 is shown in Fig. 6. With SDF, almost every point in the grid can offer corresponding information. More information brings more efficiency during training.

Here, the points outside the airfoil have positive values, and decrease as the points approaching to the boundary of the geometry where the values are zeroes, and they become negative once inside the airfoil. To generate the SDF for objects, the background mesh is an evenly distributed Cartesian grid.

3.2. CFD simulation stage

During the conventional design optimization method, computational fluid dynamics can be considered as a reliable tool for computing aerodynamic forces. To obtain the target database of our model, pressure distributions are proposed by Computational Fluids Laboratory-3D (CFL3D) [49] Navier-Stokes computer code.



(a) Structured 2D mesh. (b) Mesh around the airfoil.

Fig. 7. The computational mesh for numerical simulations.

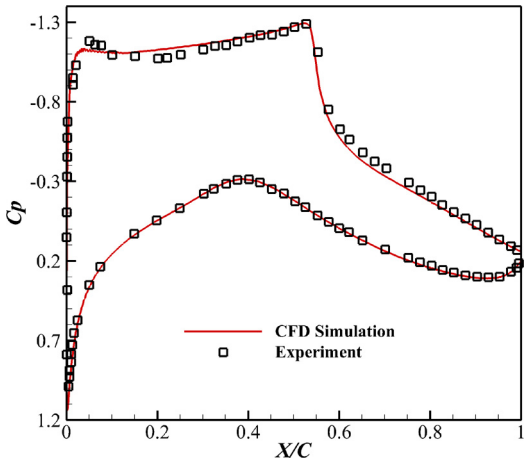


Fig. 8. Comparison between CFD simulation and experiment pressure coefficient of RAE2822 airfoil.

The profile, RAE2822, has been extensively applied for the validation of CFD solvers under the transonic flow conditions [50]. We select the flow condition of free-stream Mach number $Ma = 0.734$, free-stream Reynolds number $Re = 6.5 \times 10^6$ and angle of attack $\alpha = 2.79°$, which has experiment result in Hellstrom's work [51]. The computational mesh for numerical simulations is shown in Fig. 7.

the CFD calculation compares with wind tunnel experiment results are shown in Fig. 8. CFD solver agrees with the experiment well.

The pressure coefficient distribution is separated into two parts: the upper surface and the lower surface, then they are interpolated by 49 points on designed points along x coordinate respectively. An example of interpolating results is shown in Fig. 9. These interpolated points are revised and altered locally around the leading edge and trailing edge for fitting a curve and assuring the smoothness. Points around the shock wave ($x \in [0.4, 0.6)$) are refined especially. The distribution of interpolated points along the x coordinate is shown in Table 1.

4. Result and discussion

The network implementation, training, and testing are based on PyTorch [52]. Training and predicting are executed with precisely the same parameters on the upper and lower surface successively. The training details and prediction results are presented below.

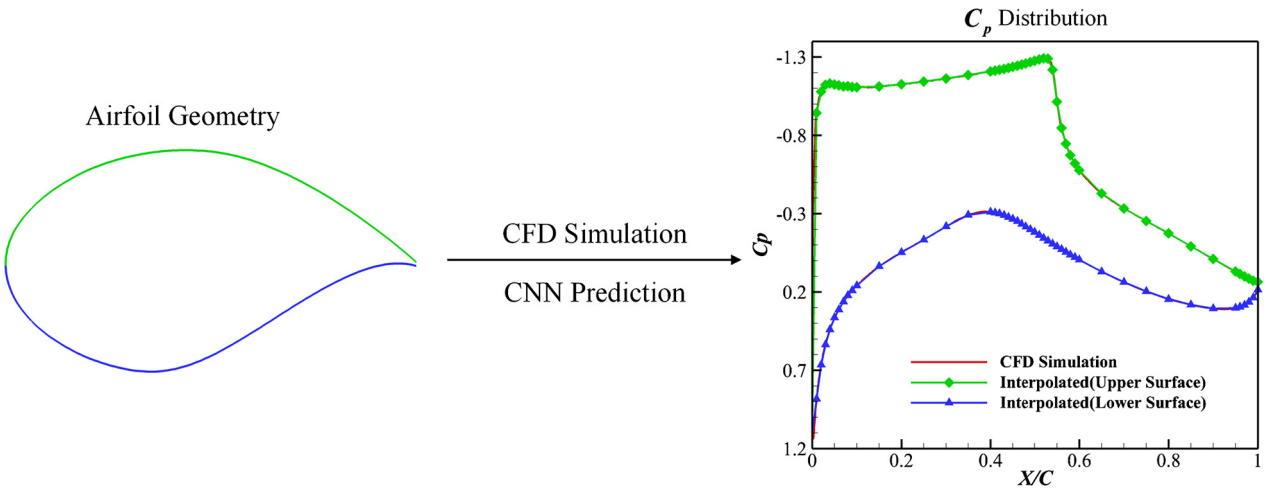


Fig. 9. Comparison between CFD simulation and interpolated pressure coefficient of RAE2822 airfoil.

Table 1
Distribution of interpolated points along the x coordinate.

Start position	End position (not included)	Interval
0	0.01	0.01
0.01	0.1	0.01
0.1	0.4	0.05
0.4	0.6	0.01
0.6	0.95	0.05
0.95	1	0.01
1	-	-

4.1. Parameters setting

We design six different neural networks, three CNNs as the experimental group, three deep forward neural networks used as the control group. Deep forward neural networks, or often known as multilayer perceptrons (MLPs), are the quintessential deep learning models. These models are called forward because the flow of the information takes place in the forward direction, the input travels through some intermediate functions in the hidden layer, which in turn is used to calculate the output. With the comparison between two kinds of neural networks, we will figure out if the CNN is more suitable for solving the prediction of airfoil walls.

The number of layers, number of nodes in each layer, kernel size, and choice of activation functions are often referred to as hyperparameters which are usually selected before training. Because of the empirical choices, parameters lead to good performances are generally unclear. Based on trial-and-error, we present three CNN architectures. These networks are investigated by varying the number of convolutional layers and BatchNorm layers. A CNN architecture with five sets of convolutional layers and BatchNorm layers is considered as the base model named CNN-C5. Then the number of convolutional layers and BatchNorm/ReLU layers sets are set to three and seven, called CNN-C3 and CNN-C7 respectively. For FNNs, the number of layers is same as CNNs, named FNN-L3, FNN-L5, and FNN-L7. The contrasts among the six networks are given in Table 2. Furthermore, all the hyperparameters, such as learning rate, batch size, and any parameters that need to be decided before training, will be consistent between CNN and FNN. The selection of the hyperparameters will be introduced in the next section.

With the efficient geometry representation and the discretization of pressure coefficient distribution, predicting pressure coefficient distribution is transformed from a regression problem, which requires predicting continuous value, into a classification problem with multiple classes. Every discretized point is a selected class. The number and the value of the chosen intervals will affect the predicting result. So far, we have the input/output pairs for CNN.

Because of the separated pressure distribution, two independent neural networks will be trained, one for the upper surface, one for the lower surface. The training input includes SDF represented airfoils from the training dataset and the corresponding discretized pressure coefficients.

4.2. Pressure coefficient prediction based on CNN

CNN training can be seen as an optimization problem, in which the parameters (weights and biases) of the network are updated or optimized using the backpropagation algorithm [53]. The optimization involves diminishing a loss function that reflects the degree of mismatch between the output predictions and the true values. The loss function of class i between ground truth y^i and predicting value $\hat{y}^i$ is given as the mean square error (MSE):

$$E_{MSE} = \frac{1}{m}\sum_{i=1}^{n}\sum_{j=1}^{m}(\hat{y}_j^i - y_j^i)^2 \tag{5}$$

where n represents batch size during training.

The training of CNN includes the optimization of weights and biases for convolutional layers (w_c, b_c), batch normalization layers (γ, β) and fully connected layer (w_f, b_f). We denote the trainable parameters as $(W, b) = ([w_c, \gamma, w_f], [b_c, \beta, b_f])$. In order to optimize these learnable parameters (W, b), one needs to obtain the error gradients $E_{(W,b)}$ with respect to trainable parameters using backpropagation algorithm. Then, a stochastic gradient descent optimizer is used to update the weights and biases as defined in the following equation:

$$(W_j^{n+1}, b_j^{n+1}) = (W_j^n, b_j^n) - \alpha\frac{\partial E_{MSE}}{\partial(W_j^n, b_j^n)} \tag{6}$$

where α is the learning rate. In this case, the learning rate is set to 0.0001; batch size is 50. One thousand airfoils are used in the training process; 500 airfoils are included in the test dataset. Also, a stochastic gradient descent optimizer called Adam [54] is used during training with its default hyper-parameters.

The SDF parametrized airfoil geometries X are fed as inputs to the CNN in order to obtain a model to predict the pressure coefficient (C_p). The airfoil geometrical features will be learned by

Table 2
Network architectures comparison among CNNs and FNNs. BN is batch normalization for short.

Network type	CNN-C3	CNN-C5	CNN-C7	FNN-L3	FNN-L5	FNN-L7
Filter size	5×5	5×5	5×5	-	-	-
1st BN Layer size	20	20	20	20	20	20
2nd BN Layer size	40	40	40	40	40	40
3rd BN Layer size	60	60	60	60	60	60
4th BN Layer size	-	80	80	-	80	80
5th BN Layer size	-	100	100	-	100	100
6th BN Layer size	-	-	120	-	-	120
7th BN Layer size	-	-	140	-	-	140
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	ReLU
FC layer size	24000	14400	2240	61440	102400	143360
Output layer size	49	49	49	49	49	49

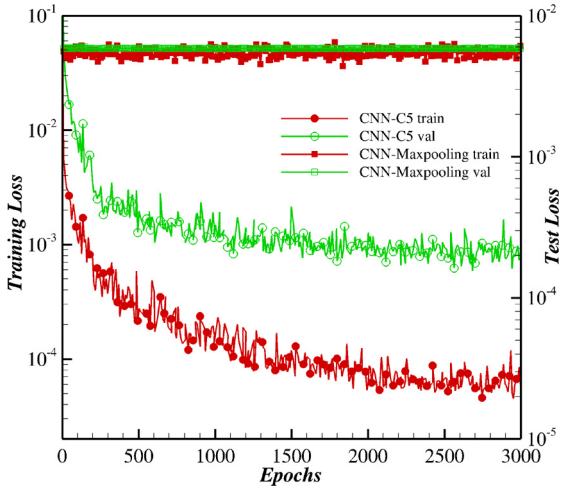


Fig. 10. Convergence history of convolution-Batch Normalization network and convolution-maxpooling network.

optimizing parameters in CNN during training. Hence, the model can be represented as:

$$C_p = f(X; W, b) \tag{7}$$

Before the results are formally presented, we give the convergence history of the convolution-Batch Normalization network and the convolution-max-pooling network to demonstrate the argument we put forward in Section 2.2.1: does the Batch Normalization improve the performances of the model? In Fig. 10, it is obvious that the convolution-max-pooling network barely optimized during the whole training. The loss stops dropping right after the training started. Thus, our model is utilized based on the convolution-Batch Normalization network.

Fig. 11 shows the convergence history of three designed CNN architectures in training and validation datasets. MSE loss is minimized around 3000 steps. Compared to the base model CNN-C5, the training and validation results of CNN-C3 are worse; and although CNN-C7 is doing well in its training process, it fails to maintain this quality during testing. From Fig. 11, changing the number of layers shows no improvements in the error convergence. It is clear that all the CNN architectures achieve similar error convergence, and hence we can consider that the hyperparameters (W, b) are converged. Among the utilized CNN architectures, the CNN-C5 architecture is considered as the base model for further study because of its steady performances.

Once the training is finished, the trained model is tested by using 500 airfoil samples, which don't participate in the training process. The MSE distribution of the prediction is shown in Fig. 12. It can be observed from the results that the model is able to achieve a less than 2% mean square error when reproducing the pressure distributions from testing airfoils. Therefore, our model is able to predict the pressure distribution of a given airfoil shape effectively. It should be noticed that there are still some outliers in the prediction, the accuracy of prediction results from upper surfaces is lower than lower surfaces, due to a relatively wrong prediction of the last point along the upper surface. During the CFD simulation phase, there could be a sudden upward rank at the trailing edge, it has no particular meaning for the pressure distribution, but its irregularity still causes some influence to our prediction.

The test accuracy bars of CNN-C5 in Fig. 13 show a statistical analysis which includes the accuracy of every point CNN predicted along the X coordinate. The accuracy is described by equation (8).

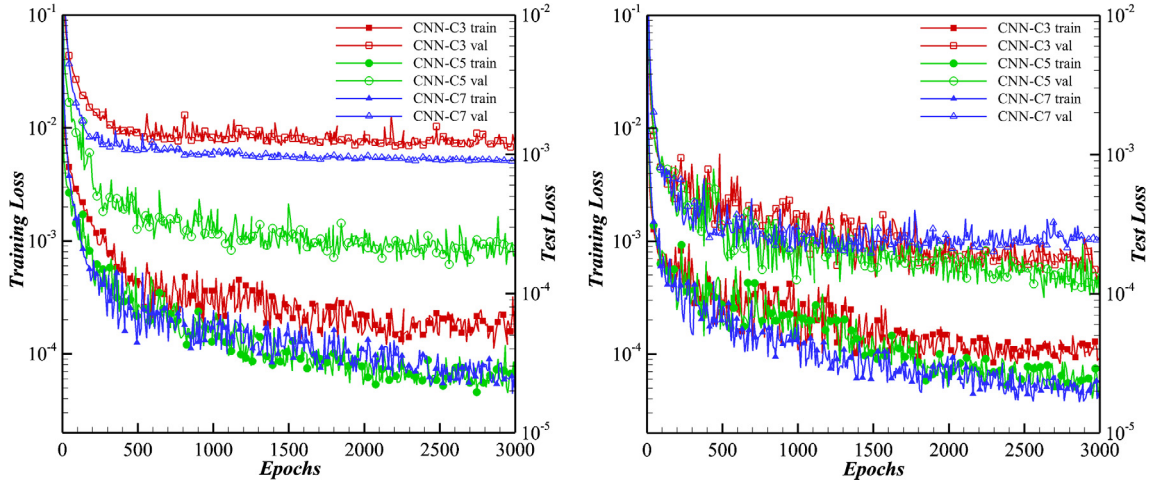
$$Accuracy = 1 - \frac{1}{n}\sum_{i=1}^{n}(\arg\max \hat{y}_j^i - \arg\max y_j^i) \tag{8}$$

In Fig. 13, the lower surface performs more steady than the upper surface, due to the similar and gentle trend of the lower surface. At the leading edge, CNN performs worse because of the variation in the leading edge along Y coordinate. Around the shock wave zone, CNN performs great, which proves that our model can capture the nonlinear features that play an important role in fluid mechanics.

Fig. 14 shows the comparison of the ground truth and the predicted pressure distribution of four randomly selected airfoils from the validation dataset. As can be seen, these four pressure forms show different features from the original airfoil RAE2822. Either a double shock structure or strong shock waves are detected for these airfoils. The trained model shows a good performance in the prediction of the pressure distribution. This confirms that the SDF method is effective, and the CNN network is able to learn the geometric features from the input images. In a word, the predicted results are reasonably accurate.

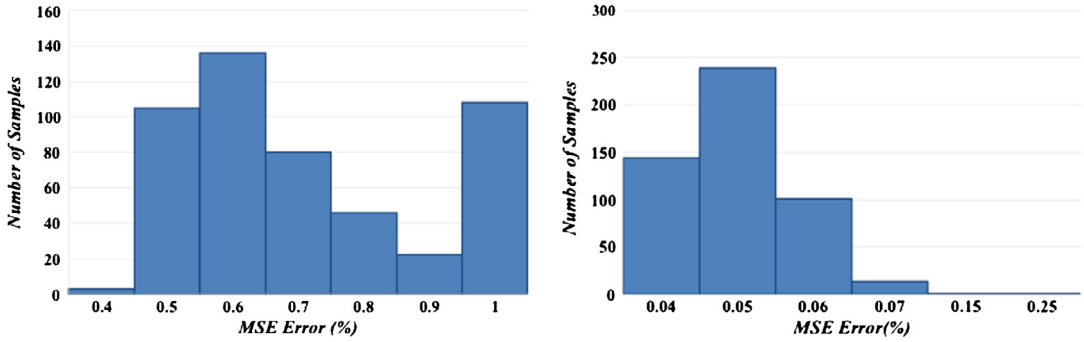
To prove the robustness of our model, we retrain and test the CNN-C5 model on a new base model: S809, whose aerodynamic characteristics are representative of wind-turbine airfoil [55]. A sketch of S809 is shown in Fig. 15. The whole process, including the data generation, parameter setting, and training method are exactly the same with case RAE2822. Thus the setup part will be skipped, the training and test results will be presented directly.

The pressure distribution of S809 (dataset does not include S809) predicted by CNN-C5 is given in Fig. 16, flow condition of free-stream Reynolds number $Re = 2 \times 10^6$ and angle of attack $\alpha = 5.13°$. Fig. 17 shows the convergence history in training and validation. The whole training trend is similar to the case using RAE2822 as the base model. Unlike RAE2822, the training of the lower surface fluctuated stronger than we expected.



(a) Convergence history of the upper surface. (b) Convergence history of the lower surface.

Fig. 11. Convergence history of the designed CNN architectures of upper and lower surface in training and validation datasets.



(a) MSE distribution of the upper surface. (b) MSE distribution of the lower surface.

Fig. 12. The MSE distribution of pressure coefficient for validation test of CNN-C5.

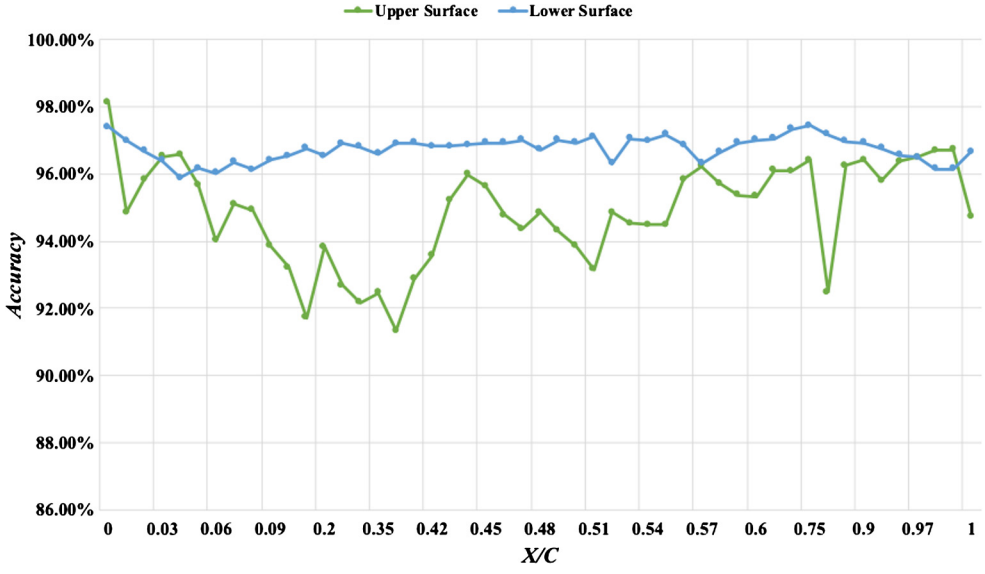


Fig. 13. Test accuracy for upper and lower surface.

Within a 2% mean square error for the validation test, we can conclude that our model successfully maps the SDF transformed airfoils into the pressure distribution with reasonable accuracy and speed. Our case provides the proof for the ability of the deep networks to learn and approximate nonlinear functions, and the efficiency of the Signed Distance Function.

4.3. Performance comparison between CNN and FNN

The performance comparison between CNN and FNN will be presented in three aspects: the training and test loss convergence history, the prediction results or accuracy, and the training time. According to the last section, the prediction of shock waves and

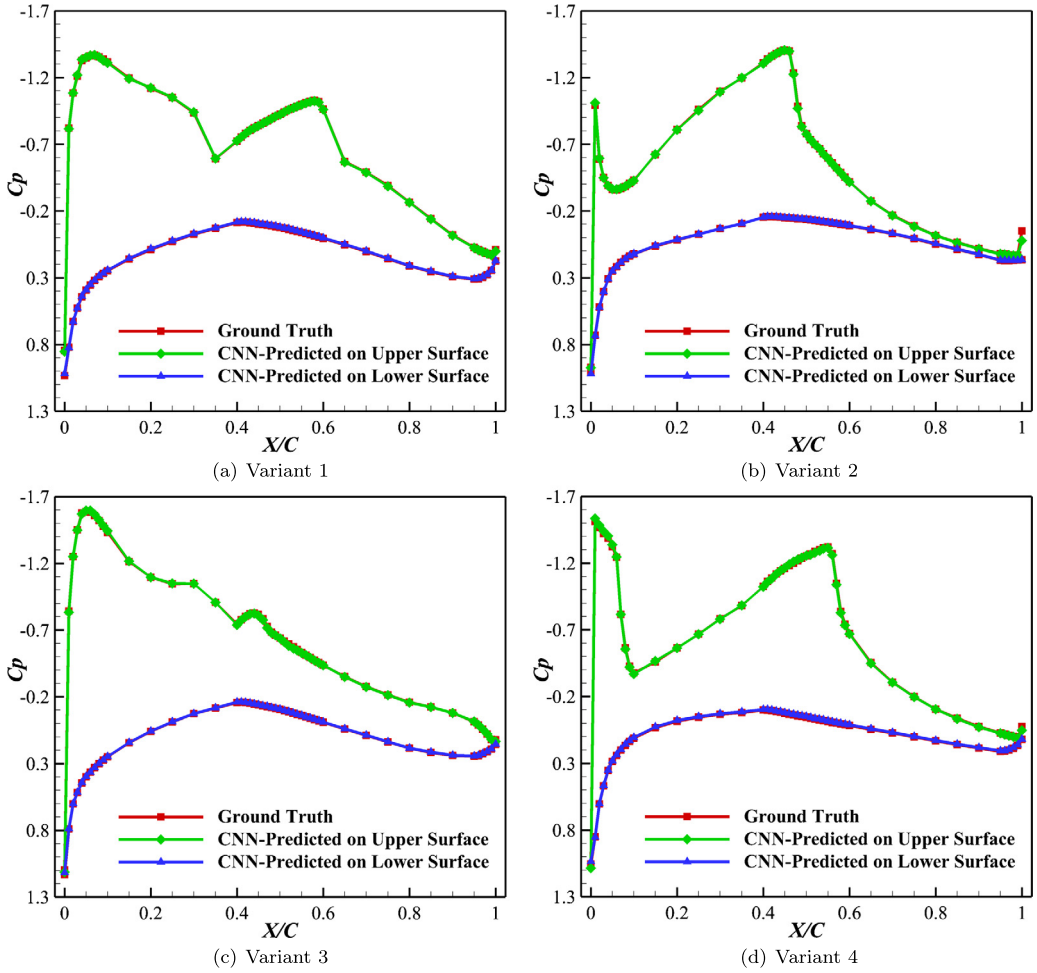


Fig. 14. Predicted pressure coefficient of random sampled airfoils of CNN-C5.

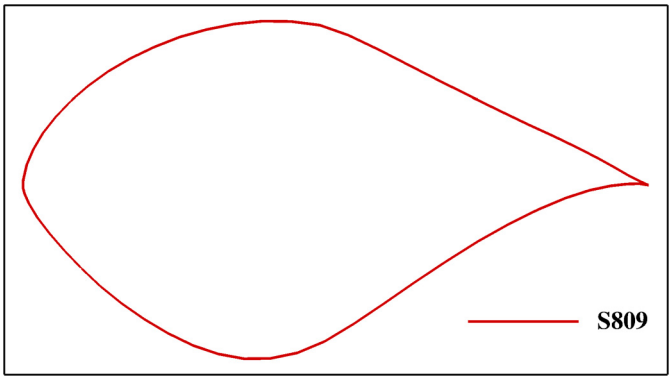


Fig. 15. A sketch of S809 airfoil.

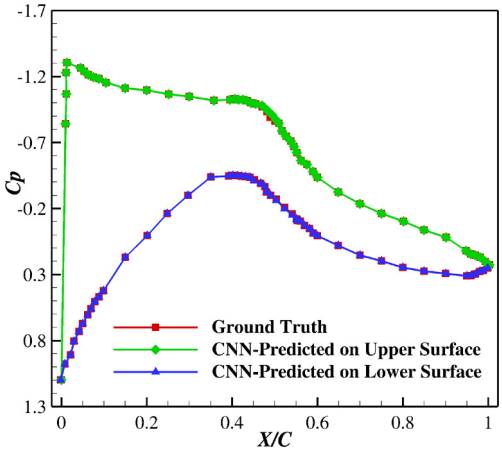
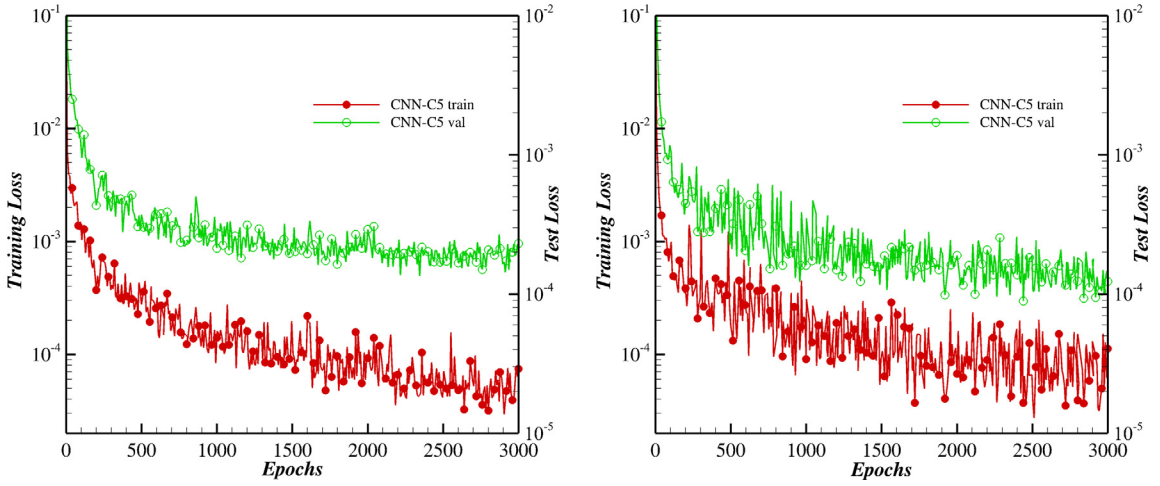


Fig. 16. Predicted pressure coefficient of S809 airfoil.

other nonlinear aerodynamic characteristics makes the training more unsteady and challenging. Thus all the results are collected based on upper surface training and test.

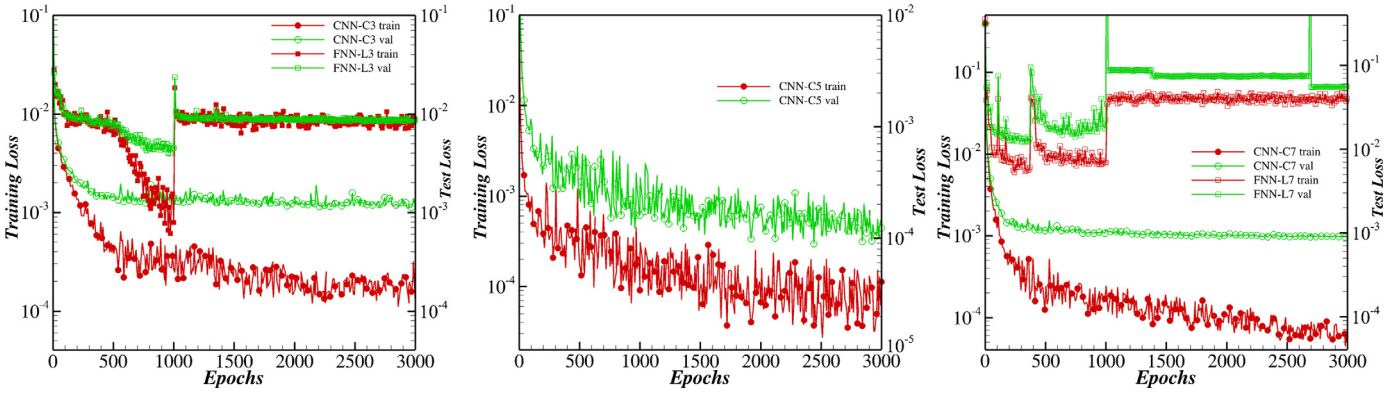
Firstly, the training process of the upper surface is shown in Fig. 18. We can see that the FNN-L3 and FNN-L5 meet a cliff fall during training. This phenomenon may cause by an oversized learning rate, which leads to a steep drop of the gradients. The FNN-L7 also encounters a sudden change of training and validation loss. Its test loss even meets some extremely high values. The losses of FNN-L3 and FNN-L5 converged to 10^{-2} around 3000 steps, but FNN-L5's loss has a strange upward. The FNN-L7's loss converged to 10^{-1}. Three FNNs all have much worse performances compare to CNNs. These suggest that the training process is unstable and may fail to predict the pressure distribution.

Next, we will verify the test accuracy. The four variants used in the last section are chosen to be the airfoil samples, their pressure distributions predicted by CNN, FNN, and CFD method at the 3000th step, are shown in Fig. 19. It is apparent that all three FNNs fail to capture the aerodynamic features, especially for FNN-L7, its model collapses cause the pressure shown in four variants are precisely the same. Shock waves are the critical information at subsonic flying conditions, the failure of describing the shock



(a) Convergence history of the upper surface. (b) Convergence history of the lower surface.

Fig. 17. Convergence history of the case base model S809 in training and validation.



(a) Convergence history of three-layer networks. (b) Convergence history of five-layer networks. (c) Convergence history of seven-layer networks.

Fig. 18. Convergence history of CNNs and FNNs in training and validation datasets of upper surface.

Table 3
Comparison of time requirements.

Operation	CPU hours (Intel @ 3.30 GHz)	GPU hours (NVIDIA GeForce)
CNN-C5 training	-	2.4
CNN-C5 prediction per case (average)	5×10^{-5}	-
FNN-L5 training	-	6.6
FNN-L5 prediction per case (average)	8×10^{-4}	-
CFD simulation time per case (average)	2.5×10^{-2}	-

waves means that FNNs are not competent enough for structured data analysis compared to CNNs.

At last, the summary of the computational time of the model training and prediction are shown in Table 3, along with the time required for CFD simulations. CNN-C5 training saves more than half of the time compared to FNN-L5. Because the number of parameters of CNN is remarkably reduced by the convolutional functions, where FNN-L5 must handle 102400 parameters when the information flows through the last layer, only 14400 parameters are considered by CNN-C5. CNN keeps accuracy and efficiency at the same time. Compared to the conventional CFD methods, CNN is about three orders of magnitudes faster and takes about only a few seconds to predict the pressure distribution once the training is completed. Overall, the proposed methodology produces fast and accurate simulation for the task of pressure prediction, and more advanced than FNN.

5. Conclusion

In this work, we presented a methodology to predict the pressure distribution (C_p) over an airfoil using deep learning techniques. This approach contains three stages, first is to convert airfoils into image data through the SDF method, which is universal and works effectively with neural networks. In the second stage, the inputs are fed into CNN. The prediction result of pressure distribution (C_p) is optimized under the assumption of $C_p = f(X; W, b)$ to train the learnable parameters (W, b). At last, the pressure distributions of untrained airfoils can be obtained directly. By utilizing both CNN and FNN, we proved that CNN has superior performances dealing with the structured data. With the further analysis of CNN, the test results show different forms of pressure distributions from the original airfoil, including a double shock structure and strong shock waves that the initial airfoil did

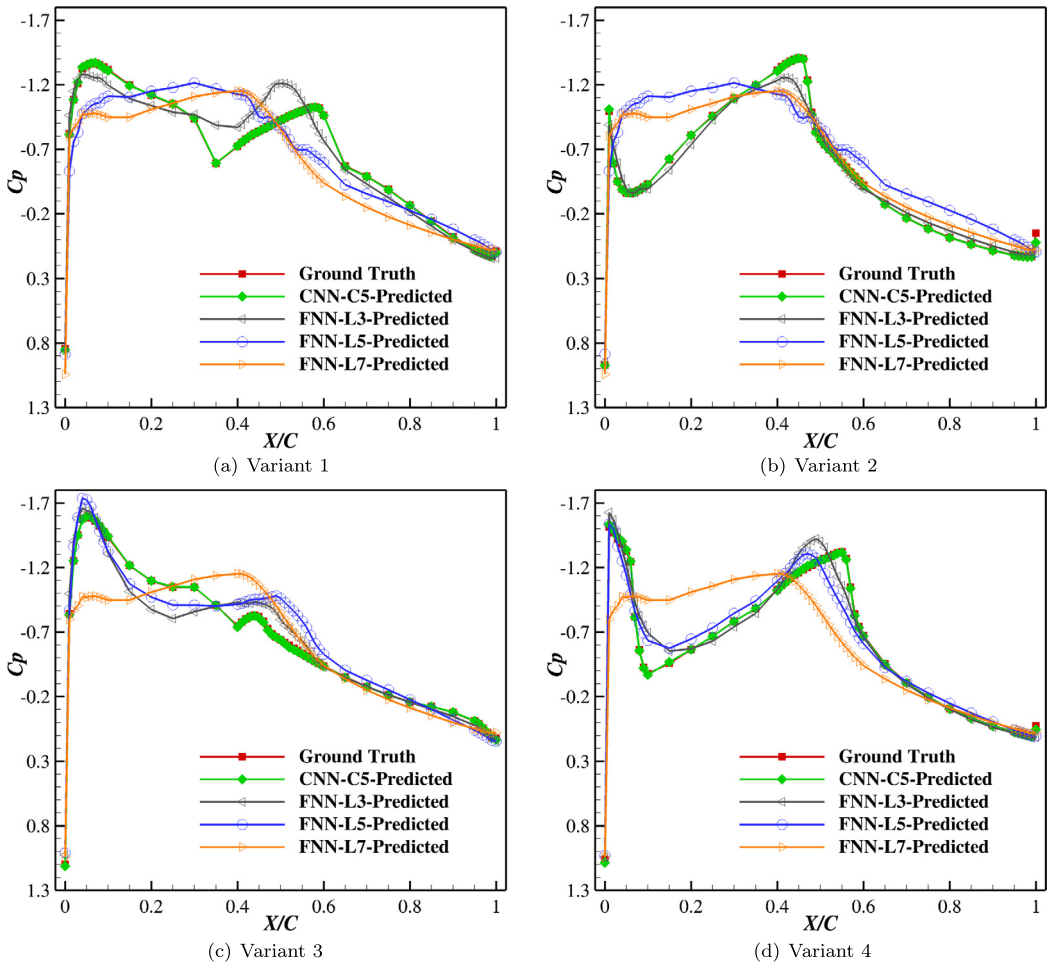


Fig. 19. Predicted pressure coefficient predicted by CNN and FNN.

not possess. As we can see from the error distribution, the presented model achieves a less than 2% mean square error of the test case. Furthermore, we list the computational time required by CFD, FNN, and CNN separately. It shows that CNN achieves three orders of magnitudes faster than CFD, and save more than half of the time compared to FNN. Hence, from the prediction results, we infer that the proposed method shows a good prediction accuracy to map the inputs into the flow variables, which proves that the deep network is capable of learning and approximating the nonlinear solutions with reasonable accuracy.

Declaration of competing interest

To the best of our knowledge and belief, neither I nor any authors have any possible conflicts of interest.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 51806178).

References

[1] B.Y. Zhou, N.R. Gauger, J. Hauth, X. Huan, M. Morelli, A. Guardone, Towards real-time in-flight ice detection systems via computational aeroacoustics and machine learning, in: AIAA Aviation 2019 Forum, 2019, p. 3103.
[2] K. Duraisamy, G. Iaccarino, H. Xiao, Turbulence modeling in the age of data, Annu. Rev. Fluid Mech. 51 (2019) 357–377.
[3] C.E. Rasmussen, Gaussian processes in machine learning, in: Summer School on Machine Learning, Springer, 2003, pp. 63–71.
[4] G. Matheron, Principles of geostatistics, Econ. Geol. 58 (8) (1963) 1246–1266.
[5] J. Sacks, W.J. Welch, T.J. Mitchell, H.P. Wynn, Design and analysis of computer experiments, Stat. Sci. (1989) 409–423.
[6] A. Chiplunkar, E. Bosco, J. Morlier, Gaussian process for aerodynamic pressures prediction in fast fluid structure interaction simulations, in: World Congress of Structural and Multidisciplinary Optimisation, Springer, 2017, pp. 221–233.
[7] X. Liu, Q. Zhu, H. Lu, Modeling multiresponse surfaces for airfoil design with multiple-output-Gaussian-process regression, J. Aircr. 51 (3) (2014) 740–747.
[8] B. Shahriari, K. Swersky, Z. Wang, R.P. Adams, N. De Freitas, Taking the human out of the loop: a review of Bayesian optimization, Proc. IEEE 104 (1) (2015) 148–175.
[9] J.-C. Jouhaud, P. Sagaut, M. Montagnac, J. Laurenceau, A surrogate-model based multidisciplinary shape optimization method with application to a 2D subsonic airfoil, Comput. Fluids 36 (3) (2007) 520–529.
[10] L. Leifsson, S. Koziel, Multi-fidelity design optimization of transonic airfoils using physics-based surrogate modeling and shape-preserving response prediction, J. Comput. Sci. 1 (2) (2010) 98–106.
[11] F. Liu, Z.-H. Han, Y. Zhang, K. Song, W.-P. Song, F. Gui, J.-B. Tang, Surrogate-based aerodynamic shape optimization of hypersonic flows considering transonic performance, Aerosp. Sci. Technol. 93 (2019) 105345.
[12] J. Tao, G. Sun, Application of deep learning based multi-fidelity surrogate model to robust aerodynamic design optimization, Aerosp. Sci. Technol. 92 (2019) 722–737.
[13] J. Ling, J. Templeton, Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty, Phys. Fluids 27 (8) (2015) 085103.
[14] J.-L. Wu, H. Xiao, E. Paterson, Physics-informed machine learning approach for augmenting turbulence models: a comprehensive framework, Phys. Rev. Fluids 3 (7) (2018) 074602.
[15] M.A. Cruz, R.L. Thompson, L.E. Sampaio, R.D. Bacchi, The use of the Reynolds force vector in a physics informed machine learning approach for predictive turbulence modeling, Comput. Fluids 192 (2019) 104258.
[16] M.M. Rai, N.K. Madavan, Aerodynamic design using neural networks, AIAA J. 38 (1) (2000) 173–182.
[17] F. Rosenblatt, The Perceptron, a Perceiving and Recognizing Automaton Project Para, Cornell Aeronautical Laboratory, 1957.

[18] T.P. Miyanawala, R.K. Jaiman, An efficient deep learning technique for the Navier-Stokes equations: application to unsteady wake flow dynamics, arXiv preprint, arXiv:1710.09099, 2017.
[19] G. Sun, Y. Sun, S. Wang, Artificial neural network based inverse design: airfoils and wings, Aerosp. Sci. Technol. 42 (2015) 415–428.
[20] Y. Lin, J.-w. Zhang, H. Liu, Deep learning based short-term air traffic flow prediction considering temporal–spatial correlation, Aerosp. Sci. Technol. 93 (2019) 105113.
[21] V. Pesce, S. Silvestrini, M. Lavagna, Radial basis function neural network aided adaptive extended Kalman filter for spacecraft relative navigation, Aerosp. Sci. Technol. 96 (2020) 105527.
[22] X. Yan, J. Zhu, M. Kuang, X. Wang, Aerodynamic shape optimization using a novel optimizer based on machine learning techniques, Aerosp. Sci. Technol. 86 (2019) 826–835.
[23] A. Mannarino, P. Mantegazza, Nonlinear aerodynamic reduced order modeling by discrete time recurrent neural networks, Aerosp. Sci. Technol. 47 (2015) 406–419.
[24] M. Habibnia, J. Pascoa, ANN assisted flow modeling and analysis for a cyclorotor in ground effect, Aerosp. Sci. Technol. 95 (2019) 105495.
[25] A. Kharal, A. Saleem, Neural networks based airfoil generation for a given cp using Bezier–PARSEC parameterization, Aerosp. Sci. Technol. 23 (1) (2012) 330–344.
[26] V. Sekar, Q. Jiang, C. Shu, B.C. Khoo, Fast flow field prediction over airfoils using deep learning approach, Phys. Fluids 31 (5) (2019) 057103.
[27] X. Guo, W. Li, F. Iorio, Convolutional neural networks for steady flow approximation, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2016, pp. 481–490.
[28] V. Sekar, M. Zhang, C. Shu, B.C. Khoo, Inverse design of airfoil using a deep convolutional neural network, AIAA J. 57 (3) (2019) 993–1003.
[29] K. Duraisamy, P.R. Spalart, C.L. Rumsey, Status, emerging ideas and future directions of turbulence modeling research in aeronautics, 2017.
[30] A.P. Singh, K. Duraisamy, Using field inversion to quantify functional errors in turbulence closures, Phys. Fluids 28 (4) (2016) 045110.
[31] M. Ma, J. Lu, G. Tryggvason, Using statistical learning to close two-fluid multiphase flow equations for a simple bubbly system, Phys. Fluids 27 (9) (2015) 092101.
[32] R. Maulik, O. San, A. Rasheed, P. Vedula, Data-driven deconvolution for large eddy simulations of Kraichnan turbulence, Phys. Fluids 30 (12) (2018) 125109.
[33] O. Hennigh, Lat-net: compressing lattice Boltzmann flow simulations using deep neural networks, arXiv preprint, arXiv:1705.09036, 2017.
[34] L. Zhu, W. Zhang, J. Kou, Y. Liu, Machine learning methods for turbulence modeling in subsonic flows around airfoils, Phys. Fluids 31 (1) (2019) 015105.
[35] J. Tompson, K. Schlachter, P. Sprechmann, K. Perlin, Accelerating Eulerian fluid simulation with convolutional networks, in: Proceedings of the 34th International Conference on Machine Learning-Volume 70, 2017, pp. 3424–3433, JMLR.org.
[36] X. Jin, P. Cheng, W.-L. Chen, H. Li, Prediction model of velocity field around circular cylinder over various Reynolds numbers by fusion convolutional neural networks based on pressure on the cylinder, Phys. Fluids 30 (4) (2018) 047105.
[37] S. Bhatnagar, Y. Afshar, S. Pan, K. Duraisamy, S. Kaushik, Prediction of aerodynamic flow fields using convolutional neural networks, Comput. Mech. 64 (2) (2019) 525–545.
[38] Z. Zuo, B. Shuai, G. Wang, X. Liu, X. Wang, B. Wang, Y. Chen, Convolutional recurrent neural networks: learning spatial dependencies for image representation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 18–26.
[39] E. Yilmaz, B. German, A convolutional neural network approach to training predictors for airfoil performance, in: 18th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, 2017, p. 3660.
[40] Y. Zhang, W.J. Sung, D.N. Mavris, Application of convolutional neural network to predict airfoil lift coefficient, in: 2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2018, p. 1903.
[41] S. Ioffe, C. Szegedy, Batch normalization: accelerating deep network training by reducing internal covariate shift, arXiv preprint, arXiv:1502.03167, 2015.
[42] S. Santurkar, D. Tsipras, A. Ilyas, A. Madry, How does batch normalization help optimization?, in: Advances in Neural Information Processing Systems, 2018, pp. 2483–2493.
[43] M. Stein, Large sample properties of simulations using Latin hypercube sampling, Technometrics 29 (2) (1987) 143–151.
[44] T.W. Sederberg, S.R. Parry, Free-form deformation of solid geometric models, ACM SIGGRAPH Comput. Graph. 20 (4) (1986) 151–160.
[45] S. Coquillart, Extended Free-Form Deformation: A Sculpturing Tool for 3D Geometric Modeling, vol. 24, ACM, 1990.
[46] M. Matsugu, K. Mori, Y. Mitari, Y. Kaneda, Subject independent facial expression recognition with robust face detection using a convolutional neural network, Neural Netw. 16 (5–6) (2003) 555–559.
[47] X. Glorot, A. Bordes, Y. Bengio, Deep sparse rectifier neural networks, in: Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, 2011, pp. 315–323.
[48] G. Cybenko, Approximation by superpositions of a sigmoidal function, Math. Control Signals Syst. 2 (4) (1989) 303–314.
[49] S.L. Krist, R.T. Biedron, C.L. Rumsey, CFL3D User's Manual (Version 5.0), 1998.
[50] W. Haase, F. Brandsma, E. Elsholz, M. Leschziner, D. Schwamborn, EUROVAL—An European Initiative on Validation of CFD Codes: Results of the EC/BRITE-EURAM Project EUROVAL, 1990–1992, vol. 42, Springer-Verlag, 2013.
[51] T. Hellstrom, L. Davidson, A. Rizzi, Reynolds stress transport modelling of transonic flow around the RAE2822 airfoil, in: 32nd Aerospace Sciences Meeting and Exhibit, 1994, p. 309.
[52] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, A. Lerer, Automatic differentiation in PyTorch, 2017.
[53] D.E. Rumelhart, G.E. Hinton, R.J. Williams, Learning representations by back-propagating errors, in: Neurocomputing: Foundations of Research, 1988.
[54] D.P. Kingma, J. Ba, Adam: a method for stochastic optimization, arXiv preprint, arXiv:1412.6980, 2014.
[55] W. Wolfe, S. Ochs, W. Wolfe, S. Ochs, CFD calculations of S809 aerodynamic characteristics, in: 35th Aerospace Sciences Meeting and Exhibit, 1997, p. 973.