**ORIGINAL PAPER**

# Prediction of aerodynamic flow fields using convolutional neural networks

Saakaar Bhatnagar[1] · Yaser Afshar[1] · Shaowu Pan[1] · Karthik Duraisamy[1] · Shailendra Kaushik[2]

## Abstract

An approximation model based on convolutional neural networks (CNNs) is proposed for flow field predictions. The CNN is used to predict the velocity and pressure field in unseen flow conditions and geometries given the pixelated shape of the object. In particular, we consider Reynolds Averaged Navier–Stokes (RANS) flow solutions over airfoil shapes as training data. The CNN can automatically detect essential features with minimal human supervision and is shown to effectively estimate the velocity and pressure field orders of magnitude faster than the RANS solver, making it possible to study the impact of the airfoil shape and operating conditions on the aerodynamic forces and the flow field in near-real time. The use of specific convolution operations, parameter sharing, and gradient sharpening are shown to enhance the predictive capabilities of the CNN. We explore the network architecture and its effectiveness in predicting the flow field for different airfoil shapes, angles of attack, and Reynolds numbers.

**Keywords** Aerodynamics · Deep learning · Convolutional neural networks · Airfoils · RANS

## 1 Introduction

With advances in computing power and computational algorithms, simulation-based design and optimization has matured to a level that it plays a significant role in an industrial setting. In many practical engineering applications, however, the analysis of the flow field tends to be the most computationally intensive and time-consuming part of the process. These drawbacks make the design process tedious, time consuming, and costly, requiring a significant amount of user intervention in design explorations, thus proving to be a barrier between designers from the engineering process.

Data-driven methods have the potential to augment [8] or replace [12] these expensive high-fidelity analyses with less expensive approximations. Learning representations

Saakaar Bhatnagar and Yaser Afshar have the Co-First/Equal authorship.

✉ Yaser Afshar
   yafshar@umich.edu

   Karthik Duraisamy
   kdur@umich.edu

[1] Department of Aerospace Engineering, University of Michigan, Ann Arbor, MI 48109, USA

[2] General Motors Global R&D, Warren, MI 48092, USA

from the data, especially in the presence of spatial and temporal dependencies, have traditionally been limited to hand-crafting of features by domain experts. Over the past few years, deep learning approaches [4,15] have shown significant successes in learning from data, and have been successfully used in the development of novel computational approaches [29–31].

Deep learning presents a fast alternative solution as an efficient function approximation technique in high-dimensional spaces. Deep learning architectures such as deep neural networks (DNNs), routinely used in data mining, are well-suited for application on big, high-dimensional data sets, to extract multi-scale features.

Deep convolutional neural networks (CNN) belong to a class of DNNs, most commonly applied to the analysis of visual imagery. Previous works [19,40,48] have illustrated the promise of CNNs to learn high-level features even when the data has strong spatial and temporal correlations. Increasing attention being received by CNNs in fluid mechanics partly originates from their potential benefit of flexibility in the shape representation and scalability for 3D and transient problems. Figure 1 illustrates the simplified layout of a typical CNN, LeNet-5 [19] applied to the handwritten digit recognition task.
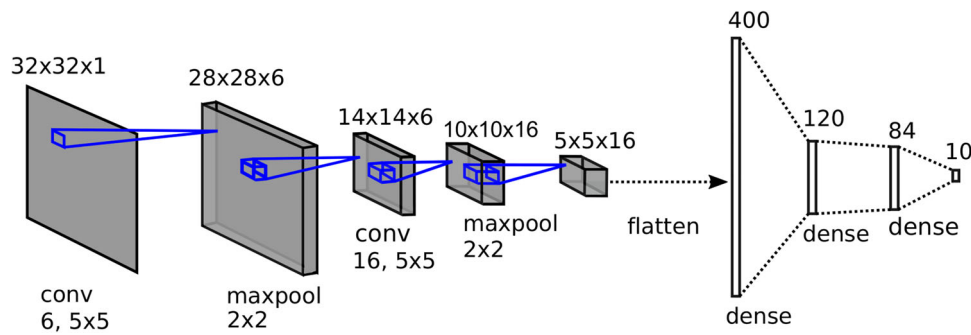
**Fig. 1** An example of a typical CNN architecture is a LeNet-5 architecture [19] in identifying handwritten digits for zip code recognition in the postal service. The architecture consists of two sets of convolutional and pooling layers. Convolutional layers use a subset of the previous layers for each filter and followed by a flattening convolutional layer, and two fully-connected layers and finally a classifier [19]
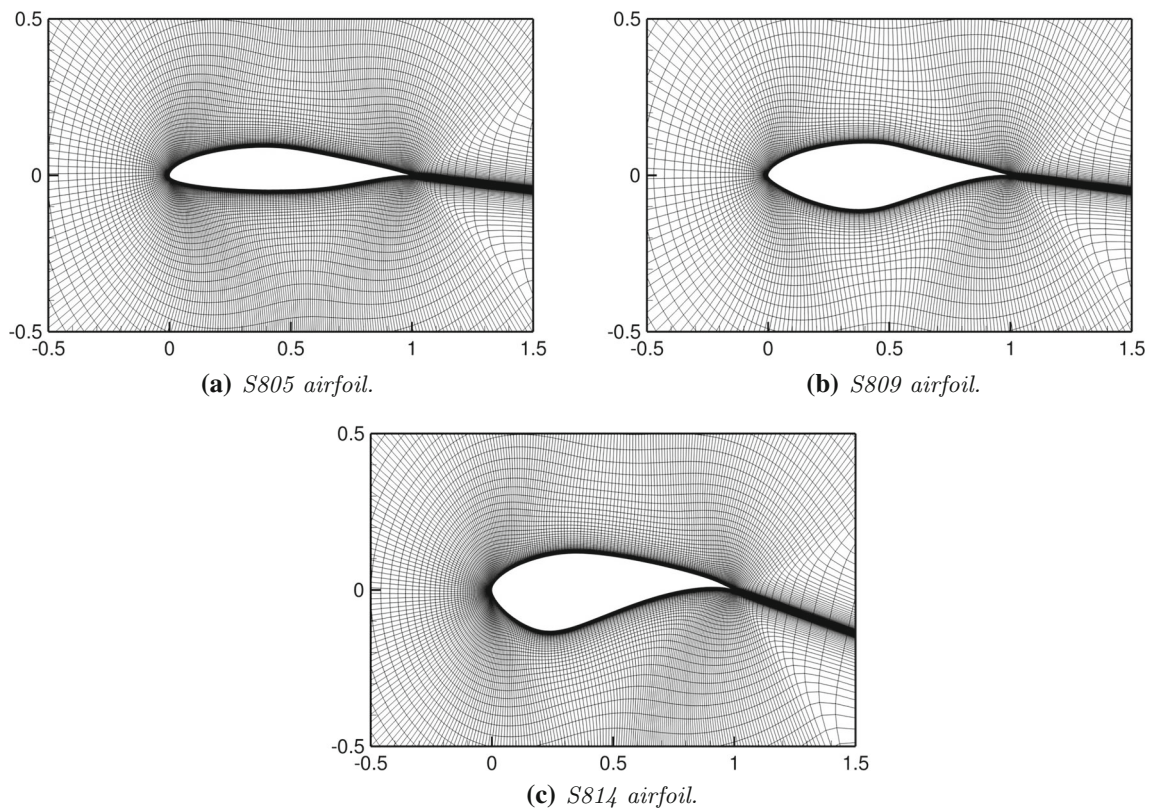


**(a)** *S805 airfoil.*



**(b)** *S809 airfoil.*



**(c)** *S814 airfoil.*

**Fig. 2** Zoomed-in view of the structured C-mesh adjacent to the airfoil surface

The main advantage of a CNN is that it exploits the low dimensional high-level abstraction by convolution. The key idea of CNN is to learn the representation and then to use a fully connected standard layer to fit the relationship between the high-level representation and the output.

## 1.1 State of the art in application of CNNs in fluid dynamics

The use of deep neural networks in computational fluid dynamics recently has been explored in several contexts.

Guo et al. [12] reported the analysis and prediction of non-uniform steady laminar flow fields around bluff body objects by employing a convolutional neural network (CNN). The authors reported a computational cost lower than that required for numerical simulations by GPU-accelerated CFD solver. Though this work was pioneering in the sense that it demonstrated generalization capabilities, and that CNNs can enable a rapid estimation of the flow field, emphasis was on qualitative estimates of the velocity field, rather than on precise aerodynamic characteristics.
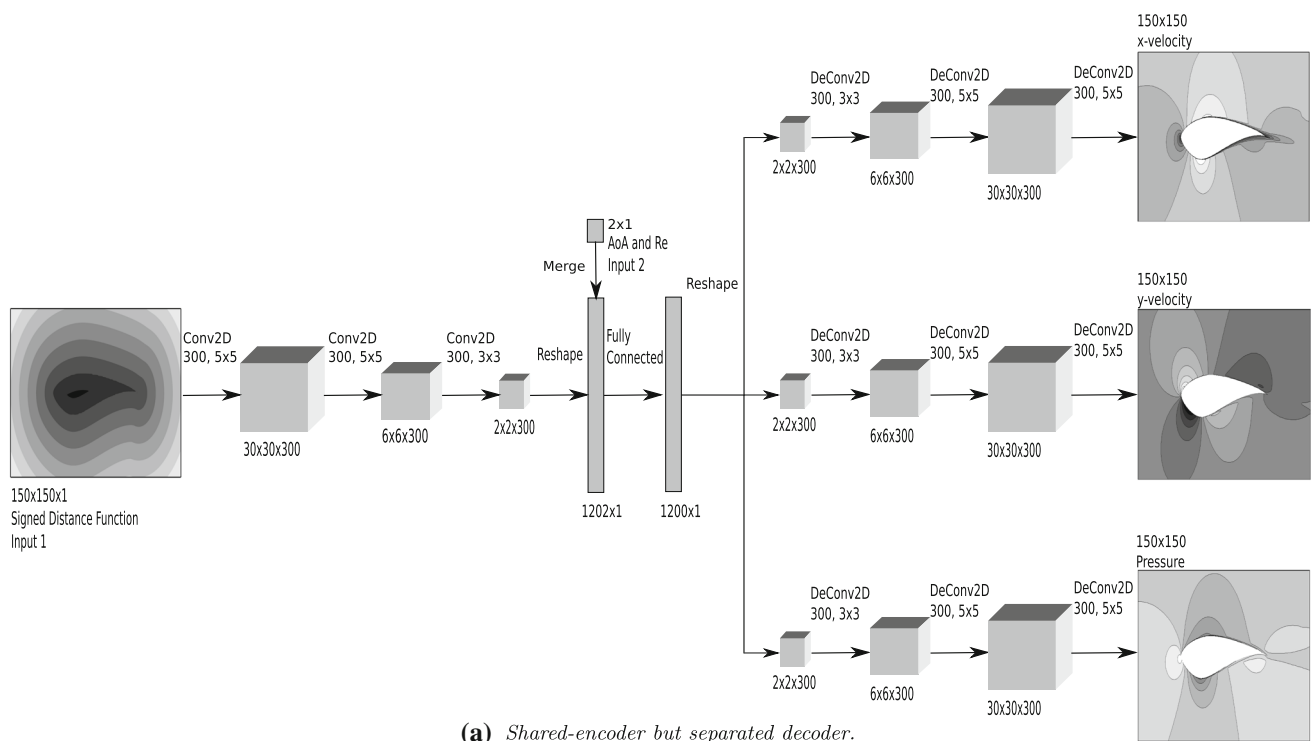
**(a)** *Shared-encoder but separated decoder.*
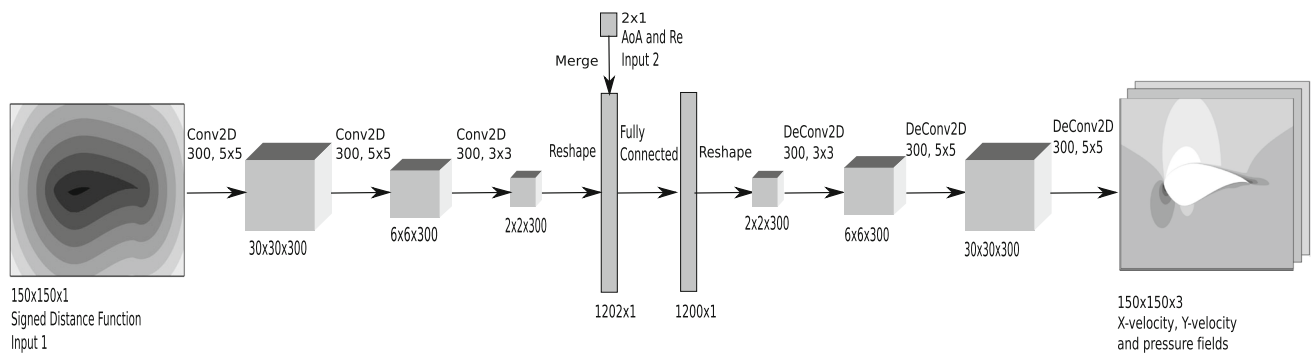


**(b)** *Shared-encoder and decoder.*

**Fig. 3** CNN based network architecture for airfoil geometry in the prediction of aerodynamic flow fields. Arrows indicate the forward operation directions. Below each operation, is the kernel size and the number of filters. The strides for convolutional and deconvolutional layers are with the size of one in each direction. Following each layer is a Swish activation function [32] except in the output layer

Miyanawala and Jaiman [25] used a CNN to predict aerodynamic force coefficients of bluff bodies at a low Reynolds number for different bluff body shapes. They presented a data-driven method using CNN and the stochastic gradient-descent for the model reduction of the Navier–Stokes equations in unsteady flow problems.

Lee and You [20,21] used a generative adversarial network (GAN) to predict unsteady laminar vortex shedding over a circular cylinder. They presented the capability of successfully learning and predicting both spatial and temporal characteristics of the laminar vortex shedding phenomenon.

Hennigh [13] presented an approach to use a DNN to compress both the computation time and memory usage of Lattice Boltzmann flow simulations. The author employed convolutional autoencoders and residual connections in an entirely differentiable scheme to shorten the state size of simulation and learn the dynamics of this compressed form.

Tompson et al. [41] proposed a data-driven approach for calculating numerical solutions to the inviscid Euler equations for fluid flow. In this approach, an approximate inference of the sparse linear system is used to enforce the Navier–Stokes incompressibility condition. This approach cannot guarantee an exact solution pressure projection step, but they showed that it empirically produces very stable divergence-free velocity fields whose runtime and accuracy
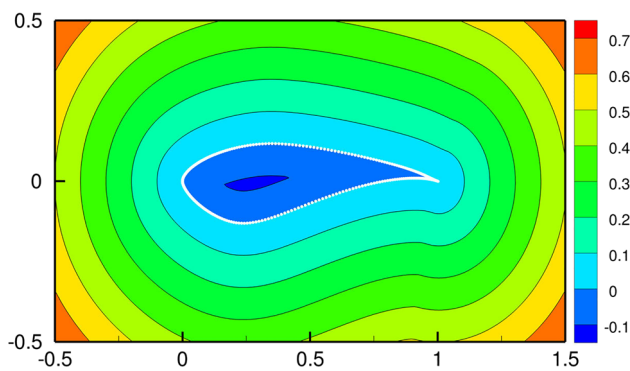
**Fig. 4** A signed distance function contour plotted for the S814 airfoil in a $150 \times 150$ Cartesian grid. The magnitude of the SDF values on the Cartesian grid equals the shortest distance to the airfoil. The airfoil boundary points are in white
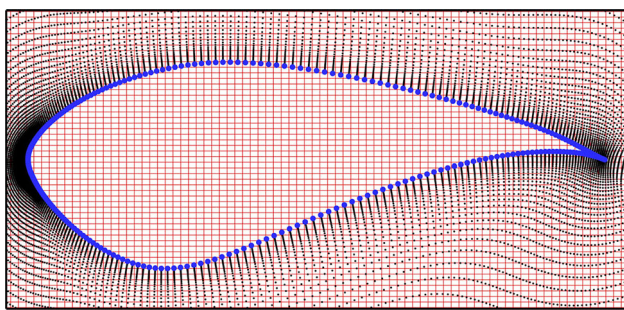


**Fig. 5** Zoomed-in view of the discrete structured C-mesh representation of an airfoil (S814) (in black) on a Cartesian grid (in red). The airfoil boundary points are in blue. (Color figure online)

is better than the Jacobi method while being orders of magnitude faster.

Zhang et al. [46] employed a CNN as feature extractor for low dimensional surrogate modeling. They presented the potential of learning and predicting lift coefficients using the geometric information of airfoil and operating parameters like Reynolds number, Mach number, and angle of attack. However, the output is not the flow field around the airfoil but the pressure coefficients at several locations. It is unclear whether this model would have good performance in predicting the drag and pressure coefficient when producing the flow field at the same time.

The primary contribution of the present work is a framework that can be used to predict the flow field around different geometries under variable flow conditions. Towards this goal and following Guo et al. [12], we propose a framework with a general and flexible approximation model for real-time prediction of non-uniform steady RANS flow in a domain based on convolutional neural networks. The flow field can be extracted from simulation data by learning the relationship between an input feature extracted from geometry and the ground truth from a RANS simulation. Then without standard convergence requirements of the RANS solver, and its
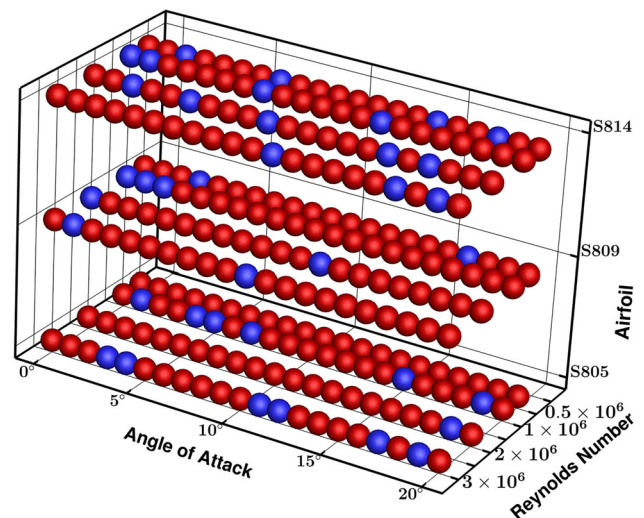


**Fig. 6** Three-dimensional scatter plots of feature space for training and testing of the network. Blue points are chosen uniformly at random as test set on the feature space, and Red points are the training set. (Color figure online)
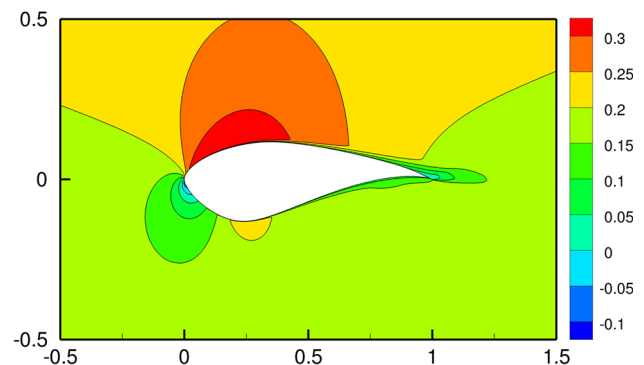


**Fig. 7** x-component of the velocity field ($U$) around S814 airfoil. RANS simulations are performed at the angle of attack of $\alpha = 9°$ and with the Reynolds number (Re) of $3 \times 10^6$
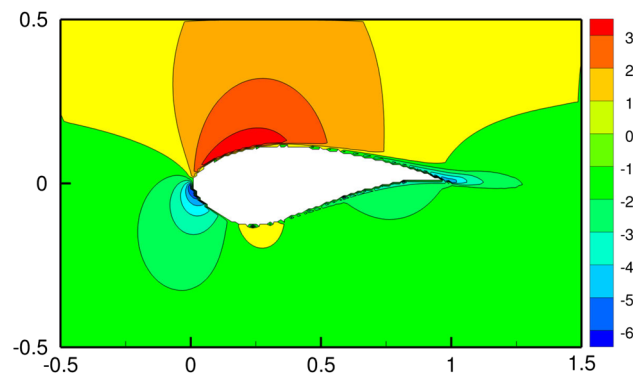


**Fig. 8** x-Component of the velocity field ($U$) around S814 airfoil interpolated from the structured C-mesh initial data in Fig. 7 onto a $150 \times 150$ Cartesian grid and normalized using the standard score normalization. Cartesian grid points inside the airfoil are set to zero
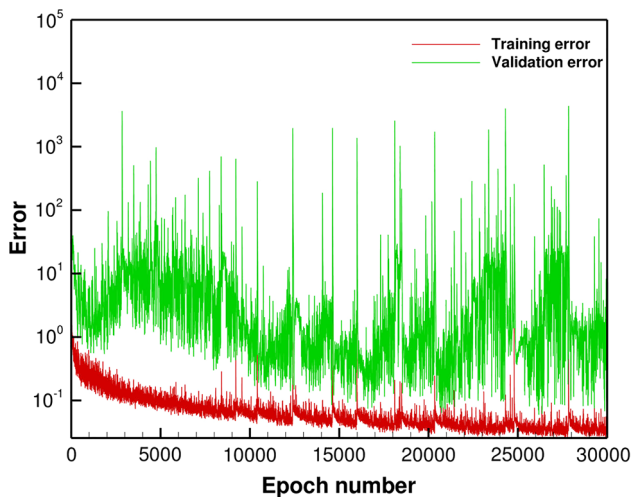
**Fig. 9** Error and learning curve of the network

number of iterations and runtime, which are irrelevant to the prediction process, we can directly predict the flow behavior in a fraction of the time. In contrast to previous studies, the present work is focused on a more rigorous characterization of aerodynamic characteristics. The present study also improves on computational aspects. For instance, Guo et al. [12] use an separated decoder, whereas the present work employs shared-encoding and decoding layers, which are computationally efficient compared to the separated alternatives.

## 2 Methodology

### 2.1 CFD simulation

In this work, flow computations and analyses are performed using the OVERTURNS CFD code [7,18]. This code solves the compressible RANS equations using a preconditioned dual-time scheme [26]. Iterative solutions are pursued using the implicit approximate factorization method [28]. Low Mach preconditioning [42] is used to improve both convergence properties and the accuracy of the spatial discretization. A third order Monotonic Upwind Scheme for Conservation Laws (MUSCL) [43] with Koren's limiter [17] and Roe's flux difference splitting [33] is used to compute the inviscid terms. Second order accurate central differencing is used for the viscous terms. The RANS closure is the SA [39] turbulence model and $\gamma - \overline{Re_{\theta t}}$ model [24] is used to capture the effect of the flow transition. No-slip boundary conditions imposed on the airfoil surface. The governing equations are provided in the Appendix.

Simulations are performed over the S805 [36], S809 [37], and S814 [38] airfoils. S809 and S814 are among a family of airfoils which contain a region of pressure recovery along the upper surface which induces a smooth transition

from laminar to turbulent flow (so-called "transition-ramp"). These airfoils are utilized in wind turbines [3]. Computations are performed using structured C-meshes with dimensions $394 \times 124$ in the wrap-around and normal directions respectively. Figure 2 shows the airfoils and their near-body meshes.

Simulations are performed at Reynolds numbers 0.5, 1, 2, and $3 \times 10^6$, respectively, and a low Mach number of 0.2 is selected to be representative of wind turbine conditions. At each Reynolds number, the simulation is performed for different airfoils with a sweep of angles of attack from $\alpha = 0°$ to $\alpha = 20°$. The OVERTURNS CFD code has been validated for relevant wind turbine applications in [3].

### 2.2 Convolutional neural networks

In this study, we consider the convolutional neural network to extract relevant features from fluid dynamics data and to predict the entire flow field in near real-time. The objective is a properly trained CNN which can construct the flow field around an airfoil in a non-uniform turbulence field, using only the shape of the airfoil and fluid flow characteristics of the free stream in the form of the angle of attack and Reynolds number. In this section, we describe the structure and components of the proposed CNN.

### 2.3 Network structure

To develop suitable CNN architectures for variable flow conditions and airfoil shapes, we build our model based on an encoder–decoder CNN, similar to the model proposed by Guo et al. [12]. Encoder–decoder CNNs are most widely used for machine translation from a source language to a target language [6]. The encoder–decoder CNN has three main components: a stack of convolution layers, followed by a dense layer and subsequently another stack of convolution layers. Figure 3 illustrates the proposed CNN architecture designed in this work.

Guo et al. [12] used a shared-encoder but separated decoder. We conjecture that the separated decoder may be a limiting performance factor. To address this issue, we designed shared-encoding and decoding layers in our configuration, which save computations compared to the separated alternatives. Explicitly, the weights of the layers of the decoder are shared where they are responsible for extracting high-level representations of pressure and different velocity components. This design provides the same accuracy of the separated decoders but, it is almost utilized 50% fewer parameters compared to the separated alternatives. Also, in the work of Guo et al. [12], the authors used only one low Reynolds number for all the experiments, but here, the architecture is trained with four high Reynolds numbers, three airfoils with different shapes and 21 different angles
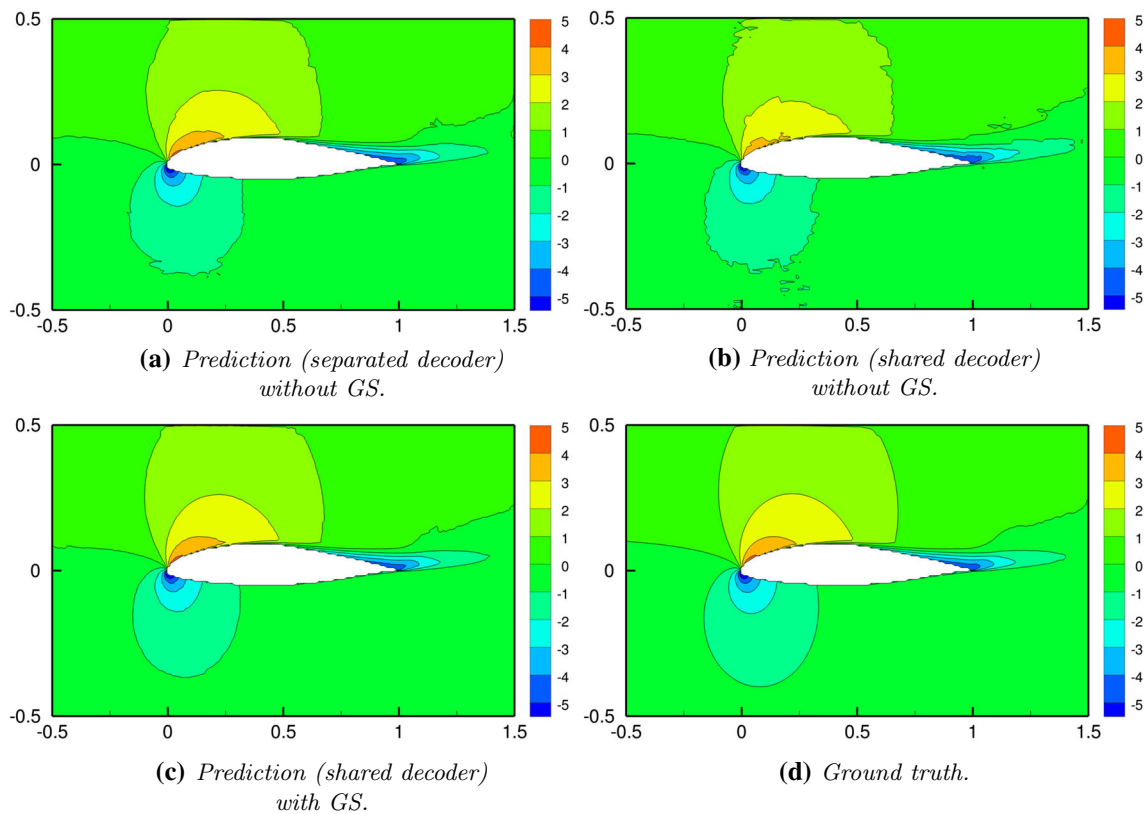
**Fig. 10** Ground truth (actual observation) versus prediction of the x-component of the velocity field around the S805 airfoil with angle of attack of $\alpha = 12.5°$ and the Reynolds number of $1 \times 10^6$. Ground truth data is interpolated from the structured C-mesh CFD results onto a $150 \times 150$ Cartesian grid and normalized using the standard score normalization

of attacks. In this architecture, we use three convolution layers both in the shared-encoding and decoding parts.

The inputs to the network are the airfoil shape and the free stream conditions of the fluid flow. We use the convolution layers to extract the geometry representation from the inputs. The decoding layers use this representation in convolution layers and generate the mapping from the extracted geometry representation to the pressure field and different components of the velocity. The network uses the Reynolds number, the angle of attack, and the shape of the airfoil in the form of $150 \times 150$ 2D array created for each data entry. The geometry representation has to be extracted from the RANS mesh and fed to the network with images. Using images in CNNs allows encoding specific properties into the architecture, and reducing the number of parameters in the network.

## 2.4 Geometry representation

A wide range of approaches are employed to capture shape details and to classify points into a learnable format. Among popular examples are methods like implicit functions in image reconstruction [5,11,14,16], or shape representation and classification [9,22,44,45]. In applications such as rendering and segmentation and in extracting structural information of different shapes, signed distance functions (SDF) are widely used. SDF provides a universal representation of different geometry shapes and represents a grid sampling of the minimum distance to the surface of an object. It also works efficiently with neural networks for shape learning. In

**Table 1** MAPE for the components of the velocity field ($U$ and $V$ respectively) and pressure in the wake region of the S805 airfoil and the entire flow field around it (separated decoder Fig. 3a)

|  | U (%) | V (%) | P (%) |
| --- | --- | --- | --- |
| Error in the wake region | 24.9 | 10.15 | 24.97 |
| Error in the entire flow | 13.51 | 11.92 | 13.50 |

**Table 2** MAPE for the components of the velocity field ($U$ and $V$ respectively) and pressure in the wake region of the S805 airfoil and the entire flow field around it (shared decoder Fig. 3b)

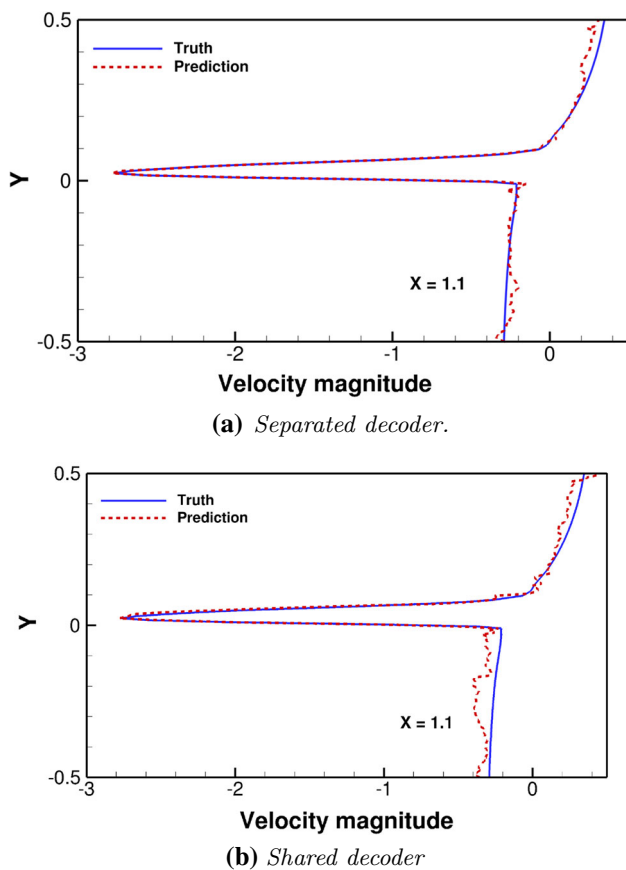|  | U (%) | V (%) | P (%) |
| --- | --- | --- | --- |
| Error in the wake region | 15.08 | 7.98 | 14.82 |
| Error in the entire flow | 9.62 | 8.65 | 7.31 |

**(a)** *Separated decoder.*



**(b)** *Shared decoder*

**Fig. 11** Prediction of the x-component velocity profile for the S805 airfoil wake at $x = 1.1$ with angle of attack of $\alpha = 12.5°$ and the Reynolds number (Re) of $1 \times 10^6$. Lower plot is for the shared decoder

this study, to capture shape details in different object representations, and following [12,27], we use the SDF sampled on a Cartesian grid. Guo et al. [12] reported the effectiveness of SDF in representing the geometry shapes for CNNs. The authors empirically showed that the values of SDF on the Cartesian grid provide not only local geometry details but also contain additional information on the global geometry structure.

## 2.5 Signed distance function

A mathematical definition of the signed distance function of a set of points $\mathbf{X}$ determines the minimum distance of each given point $\mathbf{x} \in \mathbf{X}$ from the boundary of an object $\partial \Omega$.

$$\text{SDF}(\mathbf{x}) = \begin{cases} d(\mathbf{x}, \partial \Omega) & \mathbf{x} \notin \Omega \\ 0 & \mathbf{x} \in \partial \Omega \\ -d(\mathbf{x}, \partial \Omega) & \mathbf{x} \in \Omega \end{cases}, \qquad (1)$$

where $\Omega$ denotes the object, and $d(\mathbf{x}, \partial \Omega) = \min_{\mathbf{x}_I \in \partial \Omega} (|\mathbf{x} - \mathbf{x}_I|)$ measures the shortest distance of each given point $\mathbf{x}$ from the object boundary points. The distance sign deter-

mines whether the given point is inside or outside of the object. Figure 4 illustrates the signed distance function contour plot for a S814 airfoil.

Here, the SDF has positive values at points which are outside of the airfoil, and it decreases as the point approaches the boundary of the airfoil where the SDF is zero, and it takes negative values inside the airfoil. Fast marching method [34] and fast sweeping method [47] are among the popular algorithms for calculating the signed distance function. To generate a signed distance function, we use the CFD input structured C-mesh information and define the points around the object (airfoil). Figure 5 shows the C-mesh representation of an airfoil (S814) and its boundary points on a Cartesian grid.

We find the distance of Cartesian grid points from the object boundary points, using the fast marching method [34]. To find out whether a given point is inside, outside, or just on the surface of the object, we search the boundary points and compute the scalar product between the normal vector at the nearest boundary points and the vector from the given point to the nearest one and judge the function sign from the scalar product value. For other non-convex objects, one can also use different approaches of crossing number or winding number method which are common in ray casting [10].

After pre-processing the CFD mesh files, we use the SDF as an input to feed the encoder–decoder architecture with multiple layers of convolutions. Convolution layers in the encoding-decoding part extract all the geometry features from the SDF.

## 2.6 Convolutional encoder–decoder approach

To learn all the geometry features from an input SDF, we compose the encoder and decoder with convolution layers and convolutional filters. Every convolutional layer is composed of 300 convolutional filters. Therefore, a convolution produces a set of 300 activation maps. Every convolution in our design is wrapped by a non-linear Swish activation function [32]. Swish is defined as $x \cdot \sigma(\beta x)$ where $\sigma(z) = (1 + exp(-z))^{-1}$ is the sigmoid function and $\beta$ is either a constant or a trainable parameter. The resulting activation maps are the encoding of the input in a low dimensional space of parameters to learn. The decoding operation is a convolution as well, where the encoding architecture fixes the hyper-parameters of the decoding convolution. Compared to the encoding convolution layer, here a convolution layer has reversed forward and backward passes. This inverse operation is sometimes referred to "deconvolution". The decoding operation unravels the high-level features encoded and transformed by the encoding layers and generates the mapping to the pressure field and different components of the velocity. When we use the CNN, neurons in the same feature map plane have identical weights so that the network can study concurrently, and it learns implicitly from the training data.
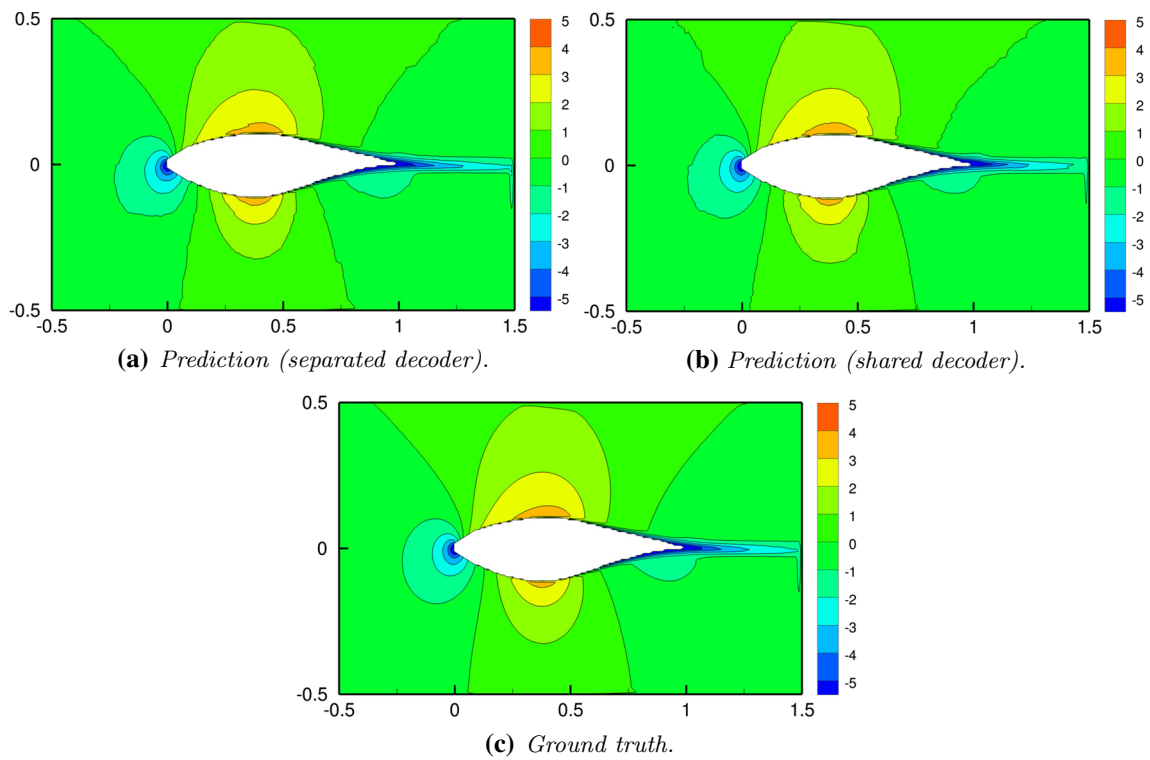
**(a)** *Prediction (separated decoder).*                    **(b)** *Prediction (shared decoder).*

**(c)** *Ground truth.*

**Fig. 12** Prediction of the x-component of the velocity field around the S809 airfoil at $\alpha = 1°$ and $Re = 1 \times 10^6$



**(a)** *Prediction*                                 **(b)** *Prediction*
*(separated decoder).*                                 *(shared decoder).*
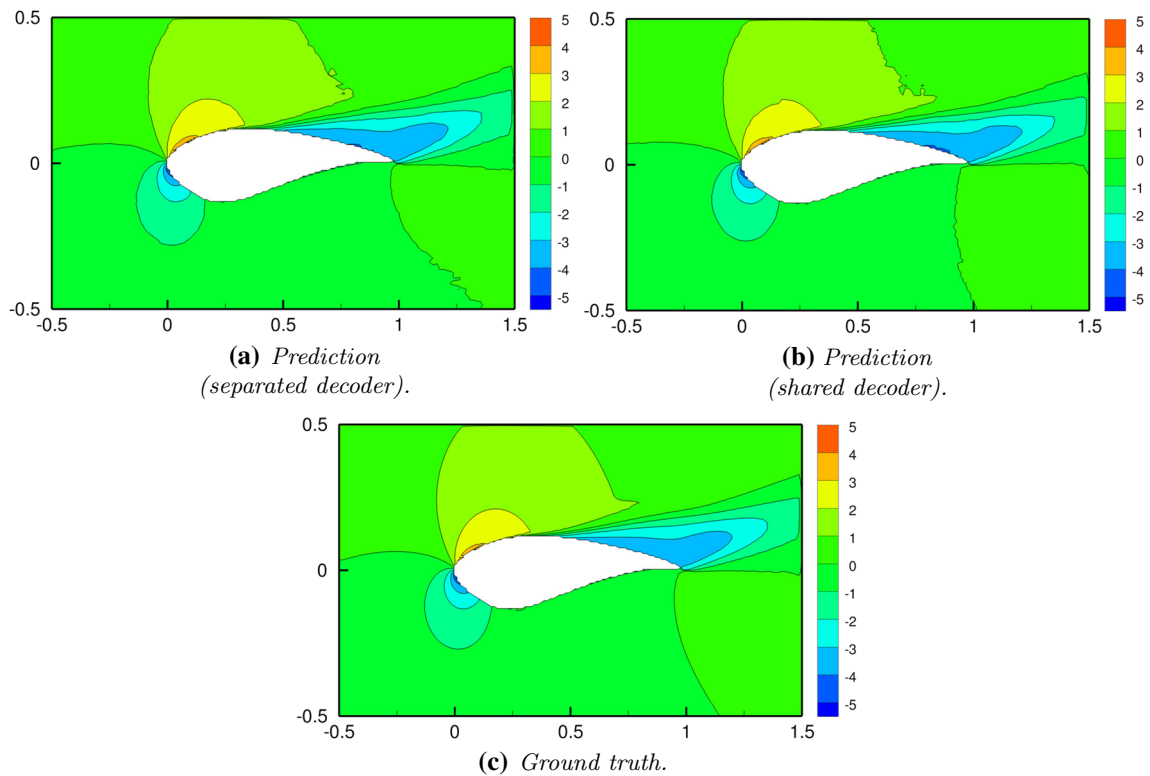
**(c)** *Ground truth.*

**Fig. 13** Prediction of the x-component of the velocity field $U$ around the S814 airfoil with $\alpha = 19°$ and $Re = 3 \times 10^6$

**Table 3** MAPE for the components of the velocity field ($U$ and $V$ respectively) and pressure (shared decoder)

| Airfoil | AOA (°) | Re $\times 10^6$ | Variable | Error in the wake region (%) | Error in the entire flow (%) |
|---------|---------|------------------|----------|------------------------------|-------------------------------|
| S809 | 1 | 1 | U | 12.25 | 10.35 |
| S809 | 1 | 1 | V | 24.27 | 11.53 |
| S809 | 1 | 1 | P | 5.14 | 8.40 |
| S814 | 19 | 3 | U | 30.80 | 13.13 |
| S814 | 19 | 3 | V | 10.43 | 5.49 |
| S814 | 19 | 3 | P | 13.84 | 5.70 |

**Table 4** MAPE for the components of the velocity field ($U$ and $V$ respectively) and pressure (separated decoder)

| Airfoil | AOA (°) | Re $\times 10^6$ | Variable | Error in the wake region (%) | Error in the entire flow (%) |
|---------|---------|------------------|----------|------------------------------|-------------------------------|
| S809 | 1 | 1 | U | 11.43 | 7.79 |
| S809 | 1 | 1 | V | 15.53 | 8.74 |
| S809 | 1 | 1 | P | 5.76 | 7.36 |
| S814 | 19 | 3 | U | 27.23 | 13.20 |
| S814 | 19 | 3 | V | 5.57 | 4.69 |
| S814 | 19 | 3 | P | 12.93 | 5.71 |

The training phase of the CNN comprises the input function, the feed-forward process, and the back-propagation process.

## 2.7 Data preparation

In total, a set of 252 RANS simulations were performed. This data includes our CFD predictions for three different S805, S809, and S814 airfoils. The training data-set consists of 85% of the full set, and the remaining data sets are used for testing, as shown in Fig. 6.

The test points are chosen uniformly at random on the feature space, providing an unbiased evaluation of a model fit on the training data-set while tuning the model's hyper-parameters.

Figure 7 shows the x-component of the velocity field ($U$) around the S814 airfoil on the structured C-mesh. The simulation is performed at an angle of attack of $\alpha = 9°$ and with the Reynolds number of $3 \times 10^6$.

The CFD data has to be interpolated onto a $150 \times 150$ Cartesian grid which contains the SDF. A triangulation-based scattered data interpolation method [2] is used. After the interpolation of the data to the Cartesian grid, the interior points are masked, and the velocity is set to zero. The comparison of the reconstructed data in Fig. 8 and the CFD data in Fig. 7 shows evidence of interpolation errors.

The interpolated data is normalized using the standard score normalization by subtracting the mean from the data and dividing the difference by the standard deviation of the data. Scaling the data causes each feature to contribute approximately proportionately to the training, and also results in a faster convergence of the network [1].

## 2.8 Network training and hyper-parameter study

The network learns different weights during the training phase to predict the flow fields. In each iteration, a batch of data undergoes the feed-forward process followed by a back-propagation (see Sect. 2.6). For a given set of input and ground truth data, the model minimizes a total loss function which is a combination of two specific loss functions and an L2 regularization as follows:

$$\text{MSE}_{\text{shared}} = \frac{1}{m(n_x - 2)(n_y - 2)} \sum_{l=1}^{m} \sum_{j=2}^{n_y-1} \sum_{i=2}^{n_x-1}$$
$$\left[ \left( U^l_{ij_{\text{truth}}} - U^l_{ij_{\text{pred}}} \right)^2 + \left( V^l_{ij_{\text{truth}}} - V^l_{ij_{\text{pred}}} \right)^2 \right.$$
$$\left. + \left( P^l_{ij_{\text{truth}}} - P^l_{ij_{\text{pred}}} \right)^2 \right], \tag{2}$$

$$\text{GS}_{\text{shared}} = \frac{1}{6m(n_x - 2)(n_y - 2)} \sum_{l=1}^{m} \sum_{j=2}^{n_y-1} \sum_{i=2}^{n_x-1}$$
$$\left[ \left( \frac{\partial P^l}{\partial x_{ij_{\text{truth}}}} - \frac{\partial P^l}{\partial x_{ij_{\text{pred}}}} \right)^2 + \left( \frac{\partial P^l}{\partial y_{ij_{\text{truth}}}} - \frac{\partial P^l}{\partial y_{ij_{\text{pred}}}} \right)^2 \right.$$
$$+ \left( \frac{\partial U^l}{\partial x_{ij_{\text{truth}}}} - \frac{\partial U^l}{\partial x_{ij_{\text{pred}}}} \right)^2 + \left( \frac{\partial U^l}{\partial y_{ij_{\text{truth}}}} - \frac{\partial U^l}{\partial y_{ij_{\text{pred}}}} \right)^2$$
$$\left. + \left( \frac{\partial V^l}{\partial x_{ij_{\text{truth}}}} - \frac{\partial V^l}{\partial x_{ij_{\text{pred}}}} \right)^2 + \left( \frac{\partial V^l}{\partial y_{ij_{\text{truth}}}} - \frac{\partial V^l}{\partial y_{ij_{\text{pred}}}} \right)^2 \right], \tag{3}$$
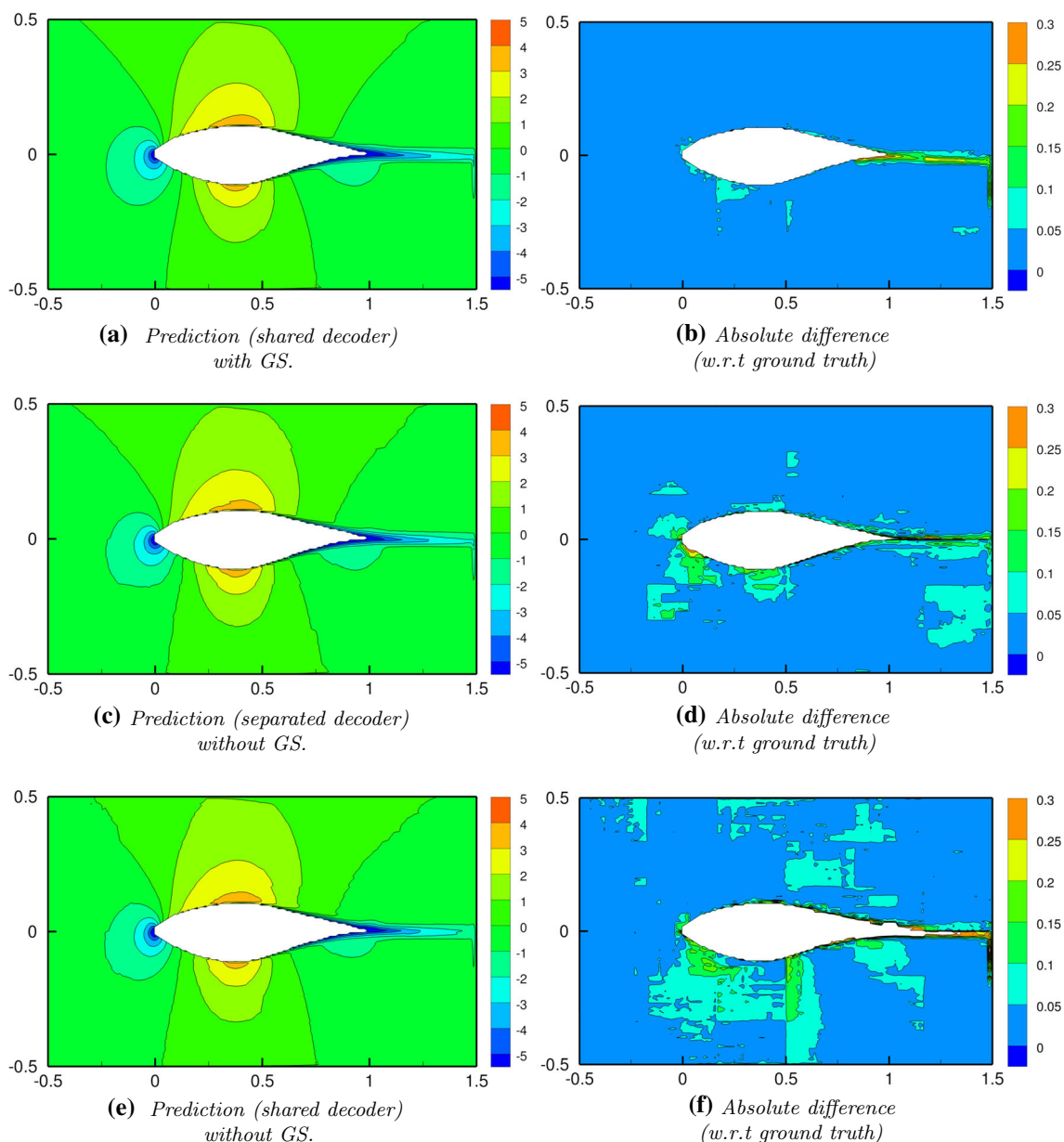
**Fig. 14** Prediction of the x-component of the velocity field $U_p$ around the S809 airfoil at $\alpha = 1°$ and $Re = 1 \times 10^6$, with and without gradient sharpening

$$\text{L2}_{\text{regularization}} = \frac{1}{2m} \sum_{l=1}^{L} \sum_{i=1}^{n_l} (\theta_i^l)^2, \tag{4}$$

where $U$, and $V$ are the x-component and y-component of the velocity field respectively, and $P$ is the scalar pressure field. $m$ is the batch size, $n_x$ is the number of grid points along the x-direction, $n_y$ is the number of grid points along the y-direction, and $L$ is the number of layers with trainable weights, and $n_l$ represents number of trainable weights in layer $l$. MSE is the mean squared error, and GS is gradient sharpening or gradient difference loss (GDL) [21,23]. In this paper, we use gradient sharpening based on a central difference operator. The network was trained for 30,000 epochs with a batch size of 214 data points, which took 33 GPU hours.
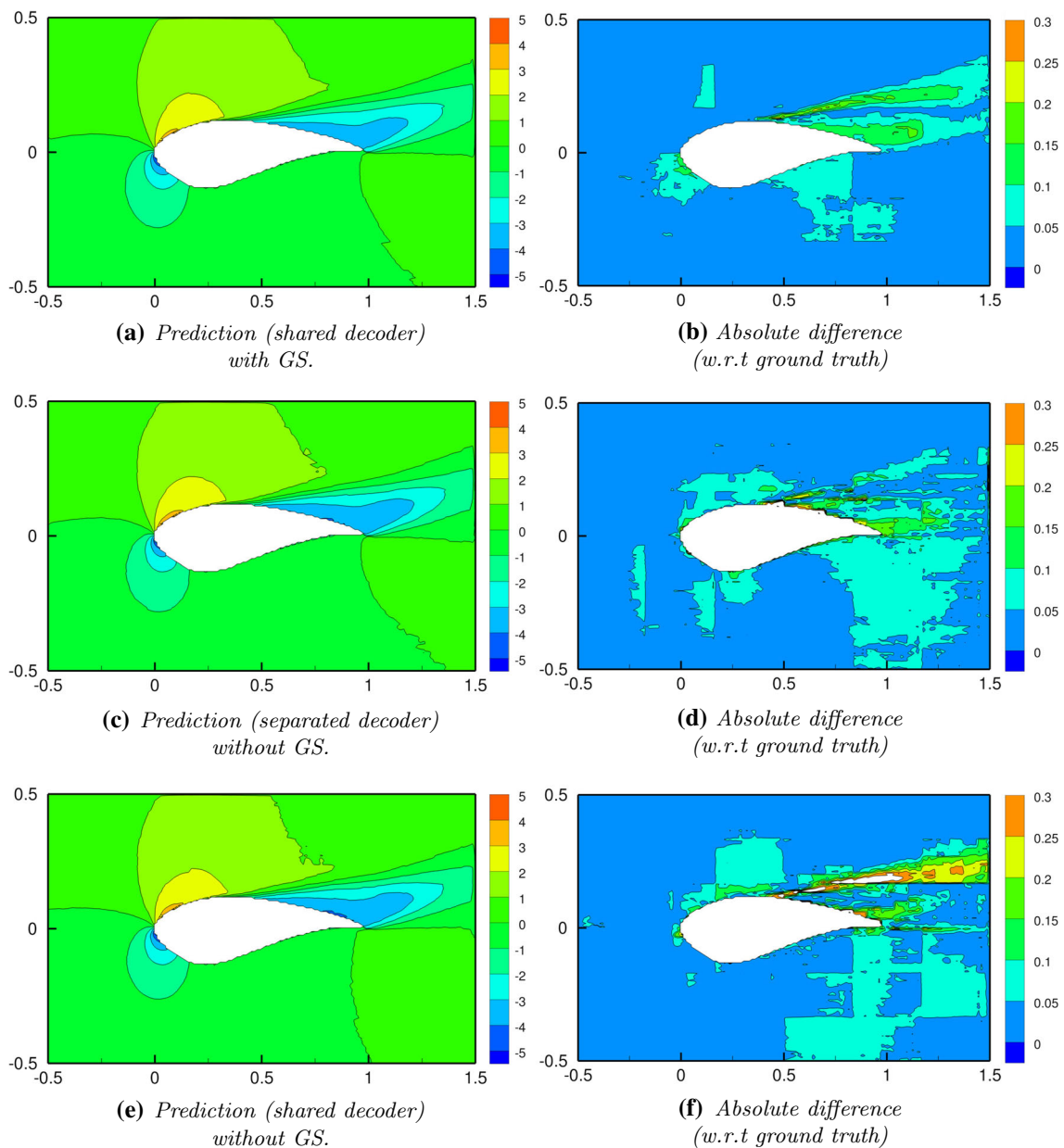
**(a)** *Prediction (shared decoder) with GS.*

**(b)** *Absolute difference (w.r.t ground truth)*

**(c)** *Prediction (separated decoder) without GS.*

**(d)** *Absolute difference (w.r.t ground truth)*

**(e)** *Prediction (shared decoder) without GS.*

**(f)** *Absolute difference (w.r.t ground truth)*

**Fig. 15** Prediction of the x-component of the velocity field $U_p$ around the S814 airfoil at $\alpha = 19°$ and $Re = 3 \times 10^6$, with and without gradient sharpening

For the separated decoder, the following loss functions are used:

$$\text{MSE}_{\text{separated}} = \frac{1}{m(n_x - 2)(n_y - 2)} \sum_{l=1}^{m} \sum_{j=2}^{n_y-1} \sum_{i=2}^{n_x-1} \left[ \left( X^l_{ij_{\text{truth}}} - X^l_{ij_{\text{pred}}} \right)^2 \right], \tag{5}$$

$$\text{GS}_{\text{separated}} = \frac{1}{2m(n_x - 2)(n_y - 2)} \sum_{l=1}^{m} \sum_{j=2}^{n_y-1} \sum_{i=2}^{n_x-1} \left[ \left( \frac{\partial X^l}{\partial x_{ij_{\text{truth}}}} - \frac{\partial X^l}{\partial x_{ij_{\text{pred}}}} \right)^2 + \left( \frac{\partial X^l}{\partial y_{ij_{\text{truth}}}} - \frac{\partial X^l}{\partial y_{ij_{\text{pred}}}} \right)^2 \right], \tag{6}$$

where $X$ stands for $U$, $V$ or $P$.

**Table 5** MAPE for the components of the velocity field ($U$ and $V$ respectively) and pressure (separated decoder)

| Airfoil | AOA (°) | Re $\times 10^6$ | Variable | Error in the wake region without GS (%) | Error in the wake region with GS (%) | Error in the entire flow without GS (%) | Error in the entire flow with GS (%) |
|---|---|---|---|---|---|---|---|
| S809 | 1 | 1 | U | 11.43 | 11.30 | 7.79 | 9.92 |
| S809 | 1 | 1 | V | 15.53 | 19.43 | 8.74 | 8.88 |
| S809 | 1 | 1 | P | 5.76 | 6.82 | 7.36 | 8.09 |
| S814 | 19 | 3 | U | 27.23 | 9.49 | 13.20 | 7.38 |
| S814 | 19 | 3 | V | 5.57 | 3.09 | 4.69 | 2.94 |
| S814 | 19 | 3 | P | 12.93 | 4.71 | 5.71 | 3.11 |
| S805 | 4 | 3 | U | 13.40 | 4.47 | 6.44 | 2.99 |
| S805 | 4 | 3 | V | 3.96 | 1.42 | 5.60 | 4.85 |
| S805 | 4 | 3 | P | 8.48 | 3.72 | 5.37 | 3.22 |

**Table 6** MAPE for the components of the velocity field ($U$ and $V$ respectively) and pressure with and without GS (shared decoder)

| Airfoil | AOA (°) | Re $\times 10^6$ | Variable | Error in the wake region without GS (%) | Error in the wake region with GS (%) | Error in the entire flow without GS (%) | Error in the entire flow with GS (%) |
|---|---|---|---|---|---|---|---|
| S809 | 1 | 1 | U | 12.25 | 8.35 | 10.35 | 5.57 |
| S809 | 1 | 1 | V | 24.27 | 17.43 | 11.53 | 6.52 |
| S809 | 1 | 1 | P | 5.14 | 3.18 | 8.40 | 3.86 |
| S814 | 19 | 3 | U | 30.80 | 19.89 | 13.13 | 11.37 |
| S814 | 19 | 3 | V | 10.43 | 3.52 | 5.49 | 3.04 |
| S814 | 19 | 3 | P | 13.84 | 7.95 | 5.70 | 3.57 |
| S805 | 4 | 3 | U | 15.68 | 15.15 | 6.50 | 9.59 |
| S805 | 4 | 3 | V | 4.20 | 1.29 | 7.63 | 10.70 |
| S805 | 4 | 3 | P | 8.58 | 32.77 | 4.03 | 15.06 |

Finding the optimal set of hyper-parameters for the network is an empirical task and is done by performing a grid search consisting of an interval of values of each hyper-parameter, and training many networks with several different combinations of these hyper-parameters. The resulting networks are compared based on generalization tendency and the difference between the truth and prediction.

# 3 Results and discussion

We first show the capability of the designed network architecture to accurately estimate the velocity and pressure field around different airfoils given only the airfoil shape. Then, we quantitatively assess the error measurement followed by a sequence of results which demonstrate usability, accuracy and effectiveness of the network.

Figure 9 illustrates the training and validation results from the network. It shows the working concept of the proposed structure, by incorporating the fluid flow characteristics and airfoil geometry. Results are presented at the epoch number with the lowest validation error.

## 3.1 Model validation

The Absolute percent error (APE) or the unsigned percentage error is used as a metric for comparison:

$$APE = \frac{|\text{Prediction} - \text{Truth}|}{|\text{Truth}|} \times 100. \qquad (7)$$

The mean value of the absolute percent error (MAPE) is standard as a Loss function for regression problems. Here, model evaluation is done using MAPE due to the very intuitive interpretation regarding the relative error and its ease of use.

In this paper, the MAPE between the prediction and the truth is calculated in the wake region of an airfoil and the entire flow field around the airfoil. Here, the wake region of the airfoil is an area defined as $\{(x, y)|x \in [1.1, 1.5], y \in [-0.5, 0.5]\}$, and $\{(x, y)|x \in [-0.5, 1.5], y \in [-0.5, 0.5]\}$ is the entire flow field area around the airfoil. The predictions
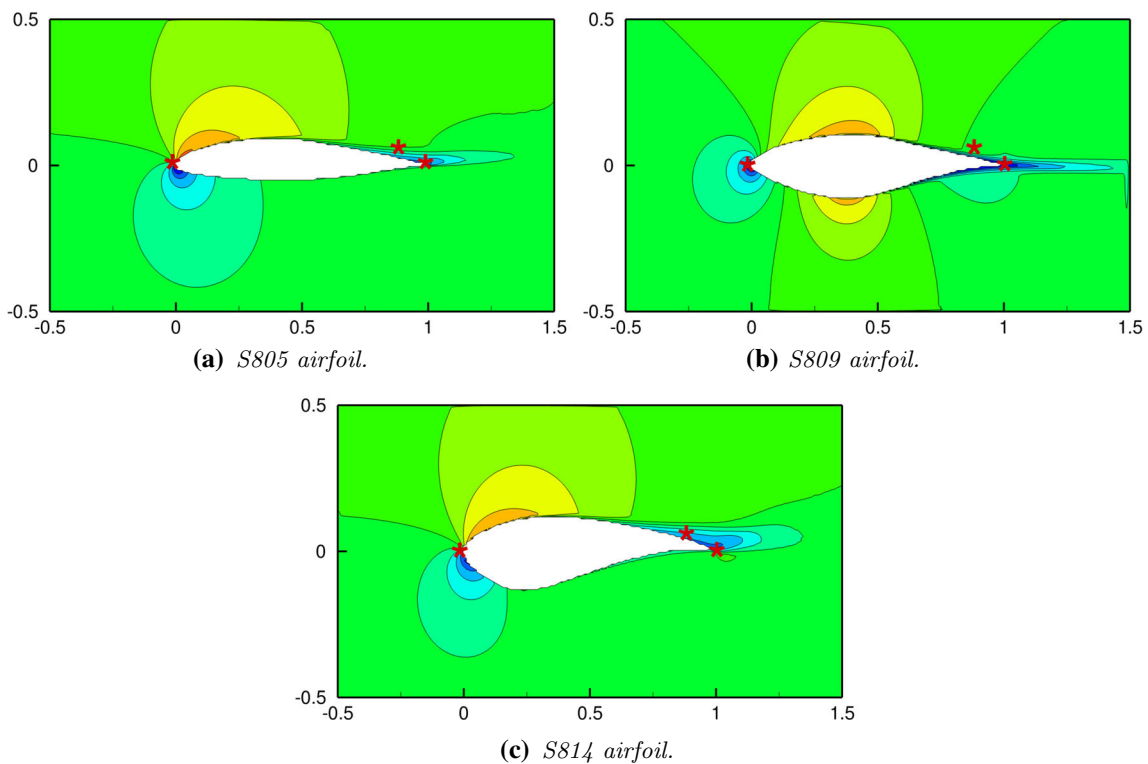
**(a)** *S805 airfoil.*



**(b)** *S809 airfoil.*



**(c)** *S814 airfoil.*

**Fig. 16** Three probes around each airfoil which are leading edge probe (LE), trailing-edge probe (TE), and the probe at the wake region of the airfoil. Each red star symbol, depicts a probe

**Table 7** APE at probe locations (shared decoder)

| Airfoil | AOA (°) | Re $\times 10^6$ | Variable | Error at LE probe (%) | Error at TE probe (%) | Error at wake probe (%) |
|---------|---------|--------|----------|----------------------|----------------------|------------------------|
| S805 | 12 | 3 | U | 2.36 | 1.67 | 31.66 |
| S805 | 12 | 3 | V | 0.96 | 0.74 | 1.50 |
| S805 | 12 | 3 | P | 9.35 | 6.56 | 22.44 |
| S809 | 1 | 3 | U | 0.08 | 2.38 | 5.45 |
| S809 | 1 | 3 | V | 0.73 | 2.22 | 1.93 |
| S809 | 1 | 3 | P | 0.23 | 3.16 | 1.07 |
| S814 | 15 | 2 | U | 0.75 | 1.98 | 1.64 |
| S814 | 15 | 2 | V | 0.76 | 1.17 | 0.18 |
| S814 | 15 | 2 | P | 5.67 | 6.56 | 8.02 |

contain 2–3% of points with an error value greater than 100%, which are treated as outliers and not included in the reported errors.

## 3.2 Numerical simulations

### 3.2.1 Angle of attack variation

At a fixed Reynolds number ($Re = 1 \times 10^6$) and fixed airfoil shape (S805), we consider simulations with angles of attack of 1° increments from $\alpha = 0°$ to $\alpha = 20°$. By using this small set of data (21 data points), we train the network with 50 filters

instead of the aforementioned 300 filters in each layer (see Sect. 2.6 for more details). The total loss function comprises only an MSE and with no regularization during training. Thus the cost function over the training set is presented as,

$$\text{Cost} = \lambda_{\text{MSE}} \times \text{MSE}, \tag{8}$$

where $\lambda_{\text{MSE}}$ is a user defined parameter (here it is $\lambda_{\text{MSE}} = 1$). After the network training is complete, testing is performed on four unseen angles of attacks, $\alpha = 2.5°$, $7.5°$, $12.5°$, and $19.5°$ respectively. Figure 10 shows the comparison between the network prediction and the actual
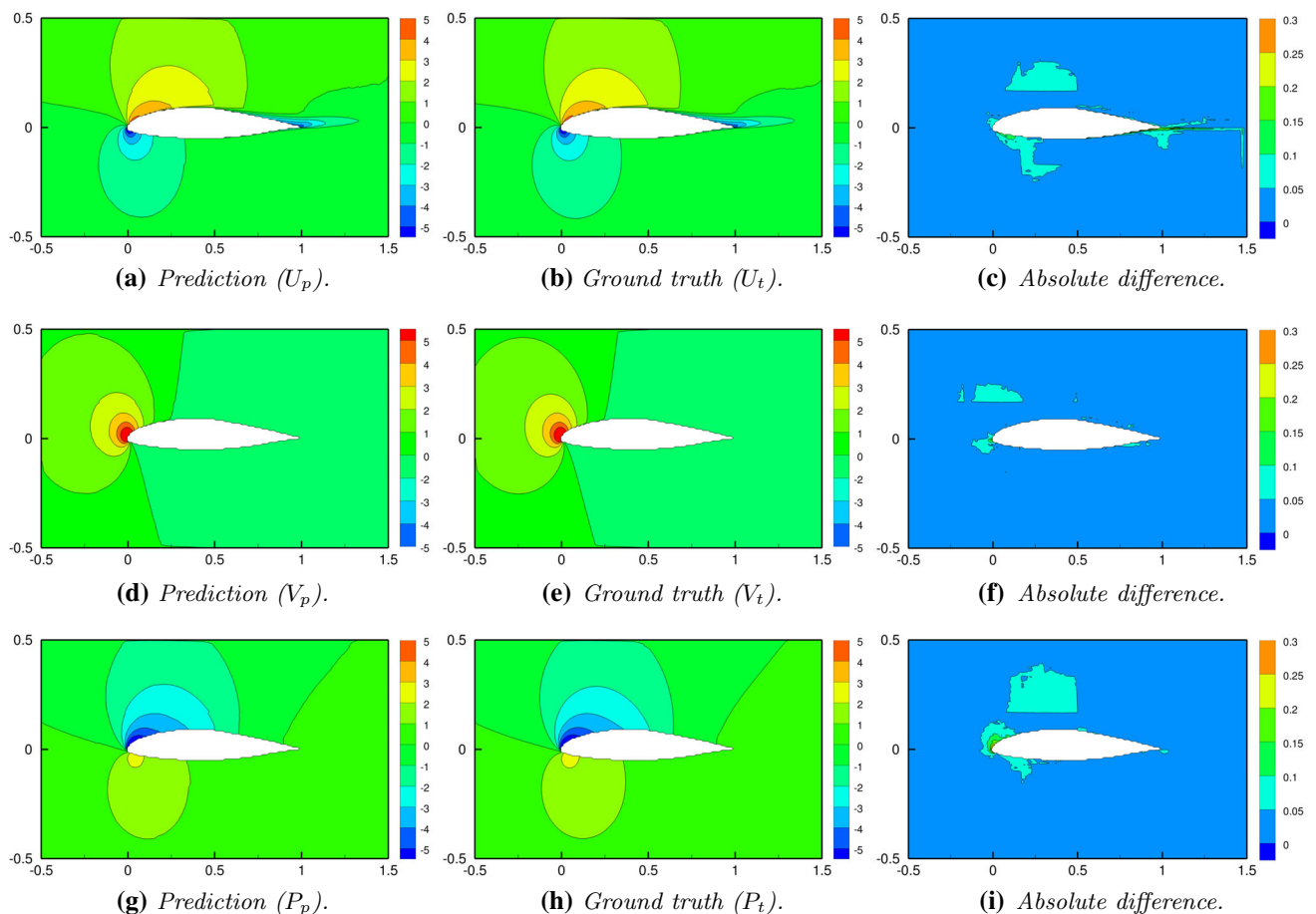
**Fig. 17** Ground truth versus prediction of the velocity field components and pressure around the S805 airfoil at $\alpha = 12°$ and $Re = 3 \times 10^6$

observation from the CFD simulation for the x-component of the velocity field around the S805 airfoil at an angle of attack of $\alpha = 12.5°$. A visual comparison shows that the prediction is in agreement with the truth.

Tables 1 and 2 present the MAPE calculated in the wake region and the entire flow field around the S805 airfoil (see Fig. 10), where the fluid flow characteristics are the angle of attack of $\alpha = 12.5°$ and the Reynolds number of $1 \times 10^6$.

The results in Tables 1 and 2, illustrate that the errors in the wake region are generally similar to the errors in the entire flow field. This trend is true not only for this case but also in subsequent experiments. Figure 11 shows the comparison between the CFD result and the network prediction of the x-component velocity profile of the airfoil wake at $x = 1.1$ (downstream location from the leading edge).

### 3.2.2 Shape, angle of attack, and Reynolds number variation

We train the network using 85% of the 252 RANS simulation data-sets, with the variation of the airfoil shape, angle of attack and Reynolds number. Every convolutional layer is

composed of 300 convolutional filters (see Sect. 2.6 for more details). The total loss function during training comprises an MSE loss function with the L2 regularization. Thus, the cost function over the training set is presented as,

$$\text{Cost} = \lambda_{\text{MSE}} \times \text{MSE} + \lambda_{\text{L2}} \times \text{L2}_{\text{regularization}}, \qquad (9)$$

where $\lambda_{\text{MSE}} = 1$ and $\lambda_{\text{L2}} = 10^{-5}$ are user defined parameters.

Figures 12 and 13 present the comparisons between the network predictions and observations for the x-component of the velocity field around the S809 and S814 airfoils at ($\alpha = 1°$, $Re = 1 \times 10^6$) and ($\alpha = 19°$, $Re = 3 \times 10^6$).

Quantitative results are presented in Tables 3 and 4.

### 3.2.3 Shape, angle of attack, and Reynolds number variation with gradient sharpening

To penalize the difference of the gradient in the loss function, and to address the lack of sharpness in predictions, we use gradient sharpening (GS) [21,23] in the loss functions combination and present the cost function over the training
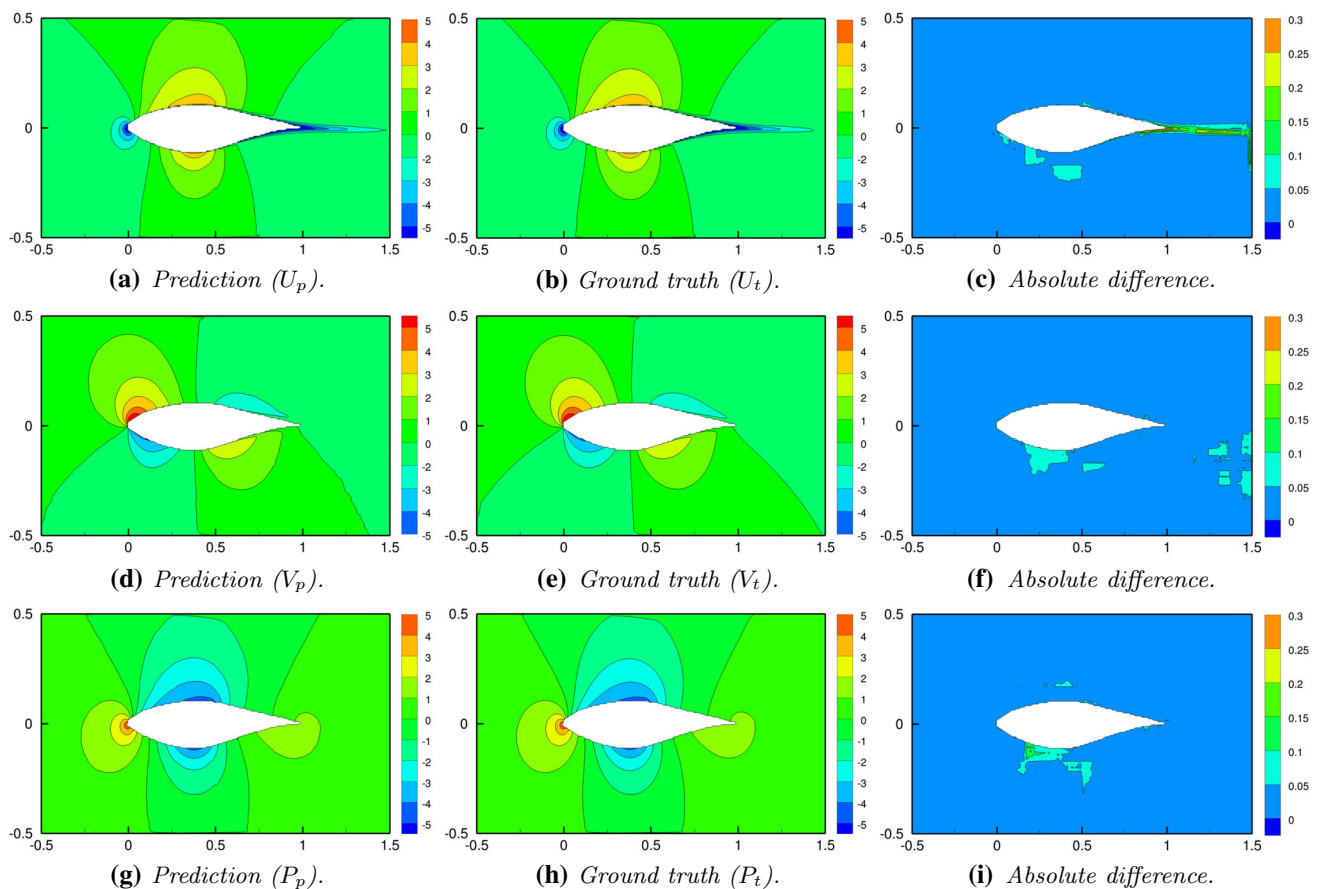
**Fig. 18** Ground truth versus prediction of the velocity field components and pressure around the S809 airfoil at $\alpha = 1°$ and $Re = 3 \times 10^6$

set as,

$$\text{Cost} = \lambda_{\text{MSE}} \times \text{MSE} + \lambda_{\text{GS}} \times \text{GS} + \lambda_{\text{L2}} \times \text{L2}_{\text{regularization}}, \quad (10)$$

where $\lambda_{\text{MSE}}$, $\lambda_{\text{GS}}$ and $\lambda_{\text{L2}}$ are the user defined parameters and their values are set via systematic experimentation, as 0.9, 0.1 and $10^{-5}$ respectively.

Figures 14 and 15 present the comparisons between the network predictions with and without GS loss for the x-component of the velocity field around S809 and S814 airfoils respectively.

Visual comparisons of the predictions and the absolute difference with and without GS as illustrated in Figs. 14 and 15 are proofs of further gains and sharpness in the network predictions. The "absolute difference" between the prediction and ground truth, for example, is defined as the absolute difference in the subtraction of each element in prediction from the corresponding element in ground truth. The MAPE for the components of the velocity field and pressure of the airfoils (S809 and S814 discussed above) are presented in Tables 5 and 6. The errors are reported in the wake region and the entire flow field around the airfoils with and without GS.

The predictions with GS in the loss function compared to not having it show significantly reduced errors in the wake region of the airfoil (20% or more in the x-component of the velocity and pressure predictions) and obvious gains and sharpness in the entire flow field around the airfoil.

To further compare the accuracy of the network predictions, we use three probes around different airfoils in different flow conditions. These probes are leading edge probe (LE), trailing-edge probe (TE), and the probe at the wake region of an airfoil. Figure 16 illustrates these three probes around different airfoils, S805, S809, and S814, respectively.

Table 7 presents the APE (Eq. 7) at the probe locations (LE, TE, and wake region probe).

Figures 17, 18 and 19 illustrate the flow-field predictions with gradient sharpening in the loss function and in comparison with the reference results from the OVERTURNS CFD code.

Figures 20 and 21 illustrate the x-component velocity profile of the airfoil wake at $x = 1.1$ (downstream location from the leading edge). These predictions include GS in the loss function.

As a further comparison of the network prediction accuracy, we consider the pressure distribution on the upper and
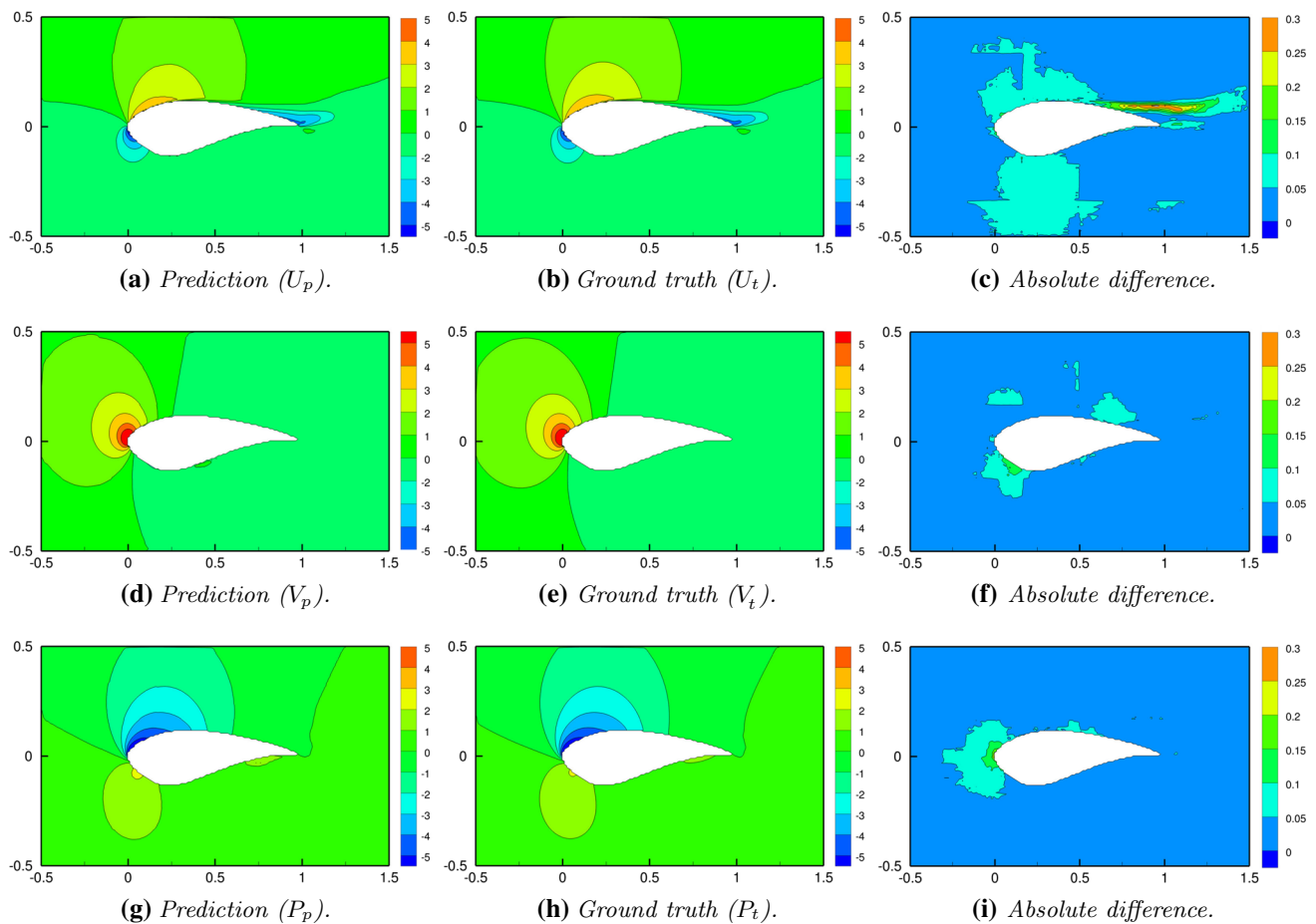
**(a)** *Prediction* ($U_p$).     **(b)** *Ground truth* ($U_t$).     **(c)** *Absolute difference.*

**(d)** *Prediction* ($V_p$).     **(e)** *Ground truth* ($V_t$).     **(f)** *Absolute difference.*

**(g)** *Prediction* ($P_p$).     **(h)** *Ground truth* ($P_t$).     **(i)** *Absolute difference.*

**Fig. 19** Ground truth versus prediction of the velocity field components and pressure around the S814 airfoil at $\alpha = 15°$ and $Re = 2 \times 10^6$

lower boundaries. Figures 22, 23 and 24 depicts the Ground truth versus Predictions of the normalized pressure using the standard score normalization along the surface of the S805, S809, and S814 airfoils respectively. It is noteworthy that the surface with a one-pixel gap adjacent to the airfoil surface is used to obtain the pressure values. This change is due to the masking of the airfoil as an input during the training.

### 3.2.4 Prediction for unseen airfoil shapes

To further explore the predictive ability and accuracy of the trained network, three unseen geometries are considered as shown in Fig. 25). The first one, denoted by "new airfoil" is an averaged shape of S809 and S814 airfoils. In addition, the S807 and S819 airfoils are also considered.

Overall, results are in good agreement with the ground truth simulation results in the entire range of angles of attacks and Reynolds numbers for the three different airfoils.

Figures 26, 27 and 28 illustrate the prediction of the network on the unseen airfoils in comparison to CFD simulations.
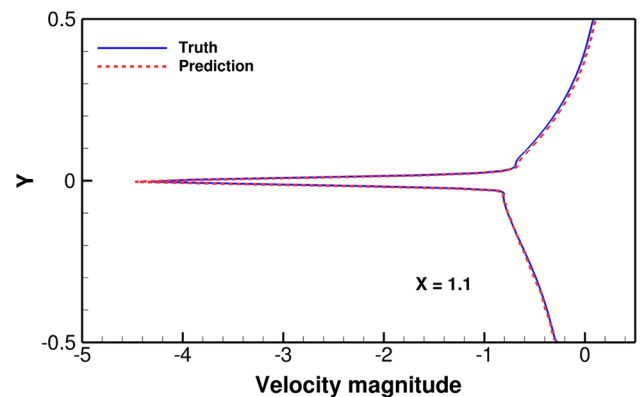


**Fig. 20** Wake velocity (x-component) prediction at $x = 1.1$ downstream of the trailing edge for the S809 airfoil at $\alpha = 1°$ and $Re = 1 \times 10^6$

Table 8 provides a quantification of the results, and suggests good generalization properties of the network to an unseen shape.
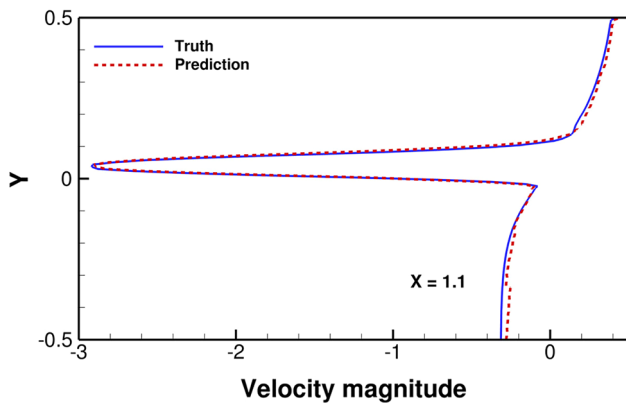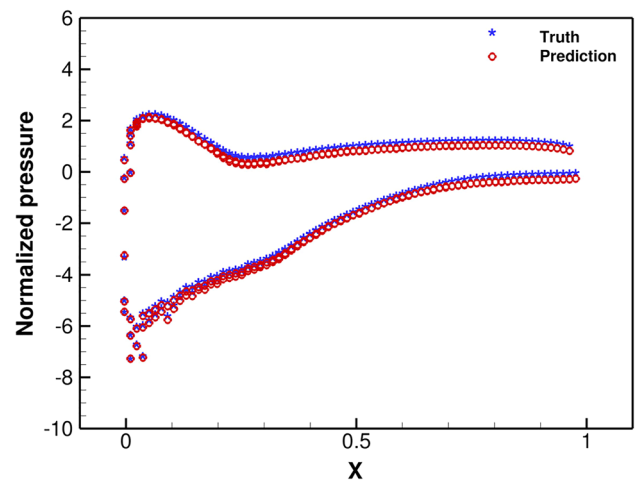
Fig. 21 Wake velocity (x-component) prediction at $x = 1.1$ downstream of the trailing edge for the S814 airfoil at $\alpha = 15°$ and $Re = 2 \times 10^6$
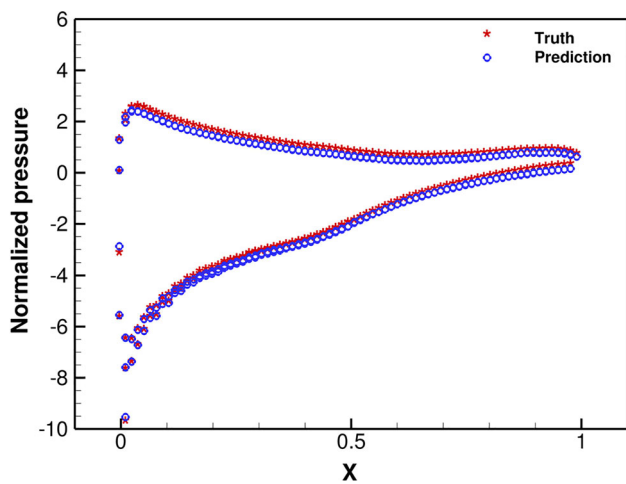


Fig. 22 Pressure prediction (standard score normalization) along the surface of the S805 airfoil at $\alpha = 12°$ and $Re = 3 \times 10^6$
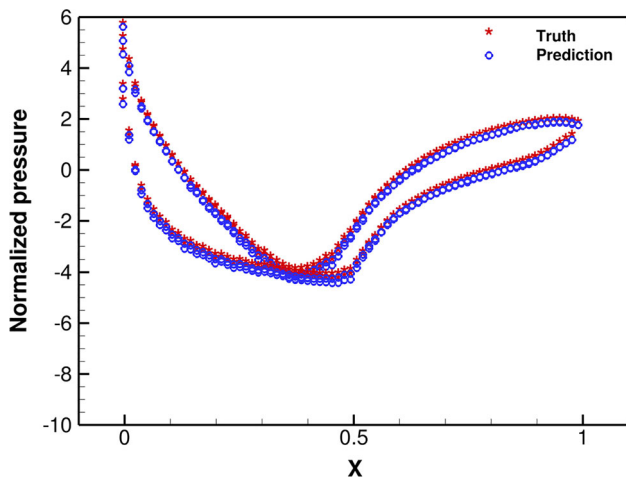


Fig. 23 Pressure prediction (standard score normalization) along the surface of the S809 airfoil at $\alpha = 1°$ and $Re = 3 \times 10^6$



Fig. 24 Pressure prediction (standard score normalization) along the surface of the S814 airfoil at $\alpha = 15°$ and $Re = 2 \times 10^6$
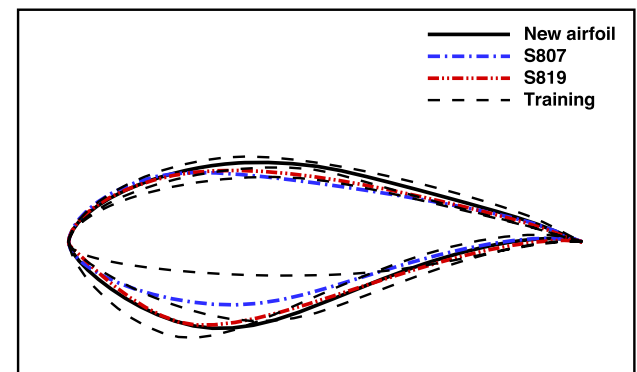


Fig. 25 Illustration of the newly created airfoil in black which is an averaged shape between the S809 and S814 airfoils respectively. The S807 and S819 airfoils are also illustrated in blue and red respectively

## 4 Conclusions and future work

A flexible approximation model based on convolutional neural networks was developed for efficient prediction of aerodynamic flow fields. Shared-encoding and decoding was used and found to be computationally more efficient compared to separated alternatives. The use of convolution operations, parameter sharing and robustness to noise using gradient sharpening were shown to enhance predictive capabilities. The Reynolds number, angle of attack, and the shape of the airfoil in the form of a signed distance function are used as inputs to the network and the outputs are the velocity and pressure fields.

The framework was utilized to predict the Reynolds Averaged Navier–Stokes flow field around different airfoil geometries under variable flow conditions. The network predictions on a single GPU were four orders of magnitude faster compared to the RANS solver, at mean square error levels of less than 10% over the entire flow field. Predictions
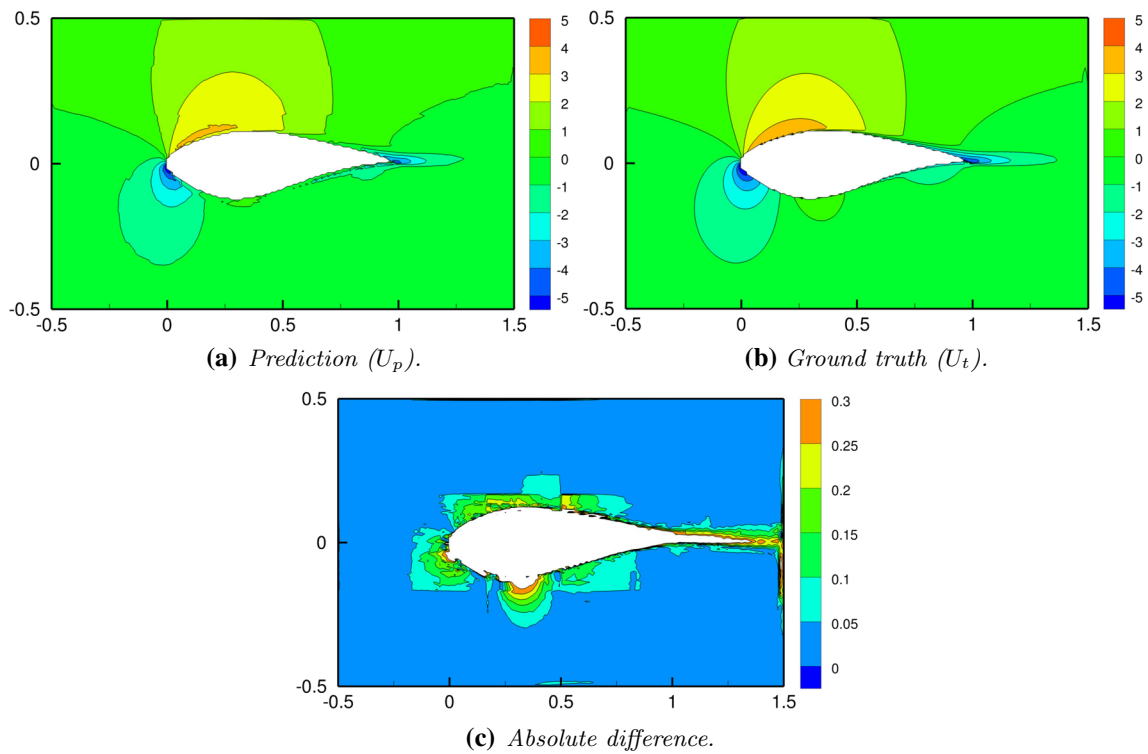
**(a)** *Prediction ($U_p$).*

**(b)** *Ground truth ($U_t$).*

**(c)** *Absolute difference.*

**Fig. 26** Ground truth versus prediction of the x-component of the velocity field ($U_t$ vs. $U_p$) and absolute difference ($|U_t - U_p|$) around the unseen airfoil at $\alpha = 9°$ and $Re = 3 \times 10^6$
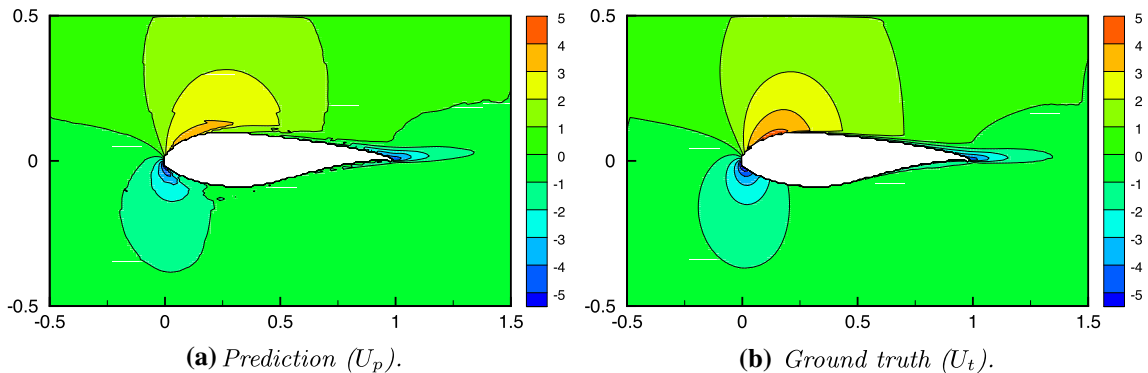


**(a)** *Prediction ($U_p$).*

**(b)** *Ground truth ($U_t$).*

**Fig. 27** Ground truth versus prediction of the x-component of the velocity field ($U_t$ vs. $U_p$) around the S807 airfoil at $\alpha = 10°$ and $Re = 1 \times 10^6$
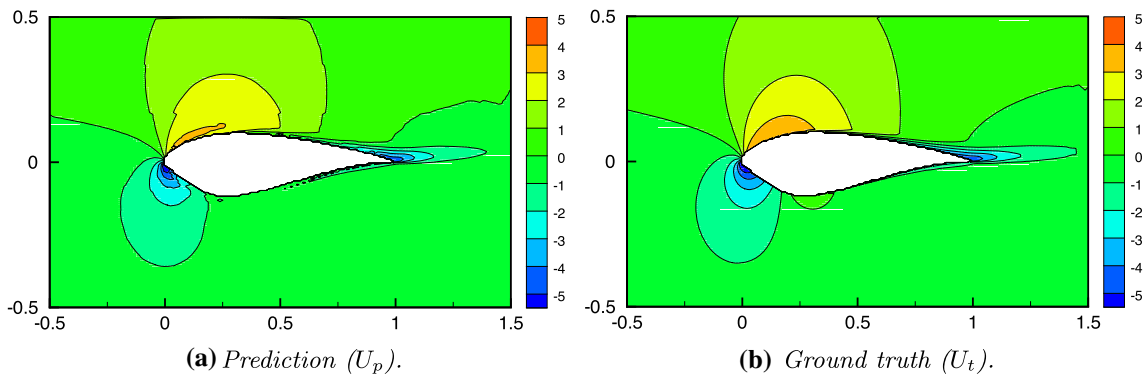


**(a)** *Prediction ($U_p$).*

**(b)** *Ground truth ($U_t$).*

**Fig. 28** Ground truth versus prediction of the x-component of the velocity field ($U_t$ vs. $U_p$) around the S819 airfoil at $\alpha = 10°$ and $Re = 1 \times 10^6$

**Table 8** MAPE for the components of the velocity field ($U$ and $V$ respectively) and pressure in the wake region of S807 and S819 airfoils respectively and the entire flow field around them

| Airfoil | Variable | Error in the wake region (%) | Error in the entire flow (%) |
| --- | --- | --- | --- |
| New | U | 10.55 | 6.41 |
| New | V | 5.71 | 8.59 |
| New | P | 4.41 | 5.27 |
| S807 | U | 11.35 | 8.4 |
| S807 | V | 2.2 | 8.8 |
| S807 | P | 6.0 | 7.6 |
| S819 | U | 13.3 | 10.3 |
| S819 | V | 2.4 | 8.6 |
| S819 | P | 5.5 | 7.5 |

were possible with a small number of training simulations, and accuracy improvements were demonstrated by employing gradient sharpening. Furthermore, the capability of the network was evaluated for unseen airfoil shapes.

The results illustrate that the CNNs can enable near real-time simulation-based design and optimization, opening avenues for an efficient design process. It is to be noted that the use of only three airfoil shapes for training is a limit factor in generalization of the predictive capabilities. Future work will seek to use a rich data set including multiple airfoil families in training and to augment the training data-sets to convert a set of input data into a broader set of slightly altered data [35] using operations such as translation and rotation. This augmentation would effectively help the network from learning irrelevant patterns, and substantially boost the performance. Furthermore, exploring physical loss functions can be helpful in explicitly imposing physical constraints such as the conservation of mass and momentum by the networks.

## Appendix: Governing equations

The RANS equations are derived by ensemble-averaging the conservation equations of mass, momentum and energy. These equations, for compressible flow are given by:

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial \left( \bar{\rho} \hat{u}_i \right)}{\partial x_i} = 0 \tag{11}$$

$$\frac{\partial \left( \bar{\rho} \hat{u}_i \right)}{\partial t} + \frac{\partial \left( \bar{\rho} \hat{u}_i \hat{u}_j \right)}{\partial x_j} = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial \bar{\sigma}_{ij}}{\partial x_j} + \frac{\partial \tau_{ij}}{\partial x_j} \tag{12}$$

$$\frac{\partial \left( \bar{\rho} \hat{E} \right)}{\partial t} + \frac{\partial \left( \bar{\rho} \hat{H} \hat{u}_j \right)}{\partial x_j} = \frac{\partial}{\partial x_j} \left( \bar{\sigma}_{ij} \hat{u}_i + \overline{\sigma_{ij} u_i''} \right)$$
$$- \frac{\partial}{\partial x_j} \left( -\hat{\kappa} \frac{\partial \hat{T}}{\partial x_j} + c_P \overline{\rho u_j'' T''} - \hat{u}_i \tau_{ij} + \frac{1}{2} \overline{\rho u_i'' u_i'' u_j''} \right), \tag{13}$$

where the overbar indicates conventional time-average mean, $u_i$ is the fluid velocity, $\rho$ is the density, $p$ is the pressure, $\tau_{ij}$ is the Reynolds stress term, $c_P$ is the heat capacity at constant pressure, and $\kappa$ is the kinetic energy of the fluctuating field (local turbulent kinetic energy). The density weighted time averaging (Favre averaging) of any quantity $\xi$, denoted by $\hat{\xi}$ is given as $\hat{\xi} = \overline{\rho \xi}/\bar{\rho}$, where,

$$\hat{H} = \hat{E} + \frac{\bar{p}}{\bar{\rho}}, \tag{14}$$

$$\bar{\sigma}_{ij} = \mu_t \left( \frac{\partial \hat{u}_i}{\partial x_j} + \frac{\partial \hat{u}_j}{\partial x_i} - \frac{2}{3} \frac{\partial \hat{u}_k}{\partial x_k} \delta_{ij} \right), \tag{15}$$

$$\tau_{ij} = -\overline{\rho u_i'' u_j''}, \tag{16}$$

$$k = \frac{\widehat{u_i''^2} + \widehat{v_i''^2} + \widehat{w_i''^2}}{2}, \tag{17}$$

$$\bar{p} = (\gamma - 1) \bar{\rho} \left[ \hat{E} - \frac{\hat{u}^2 + \hat{v}^2 + \hat{w}^2}{2} - k \right]. \tag{18}$$

To provide closure to the above equations, we use the model proposed by Spalart and Allmaras [39]. In this closure, the Boussinesq hypothesis relates the Reynolds stress and the effect of turbulence as an eddy viscosity $\mu_t$. Employing the Boussinesq approach, and Reynolds Analogy a transport equation for a working variable $\tilde{v}$ is solved to estimate the eddy viscosity field at every iteration.

$$\frac{\partial \tilde{v}}{\partial t} + u_j \frac{\partial \tilde{v}}{\partial x_j} = C_{b1} \left[ 1 - f_{t2} \right] \tilde{S} \tilde{v}$$
$$+ \frac{1}{\sigma} \left\{ \nabla \cdot \left[ (v + \tilde{v}) \nabla \tilde{v} \right] + C_{b2} |\nabla \tilde{v}|^2 \right\}$$

$$-\left[C_{w1}f_w - \frac{C_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2. \tag{19}$$

The turbulent eddy viscosity is computed as $\mu_t = \bar{\rho}\tilde{\nu}f_{v1}$, where,

$$f_{v1} = \frac{\chi^3}{\chi^3 + C_{v1}^3}, \ \chi = \frac{\tilde{\nu}}{\nu}, \ \nu = \frac{\mu}{\bar{\rho}},$$

$$f_{t2} = C_{t3}\exp\left(-C_{t4}\chi^2\right),$$

$$\tilde{S} = S + \frac{\tilde{\nu}}{\kappa^2 d^2}f_{v2},$$

$$S = \sqrt{2\Omega_{ij}\Omega ij}, \ f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}},$$

$$f_w = g\left[\frac{1 + C_{w3}^6}{g^6 + C_{w3}^6}\right]^{1/6},$$

$$g = r + C_{w2}(r^6 - r), \ r = \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2},$$

$$C_{w1} = \frac{C_{b1}}{\kappa^2} + \frac{1 + C_{b2}}{\sigma},$$

$$C_{b1} = 0.1355, \ \sigma = 2/3, \ C_{b2} = 0.622,$$

$$\kappa = 0.41, \ C_{w2} = 0.3, \ C_{w3} = 2.0, \ C_{v1} = 7.1,$$

$$C_{t3} = 1.2, \ C_{t4} = 0.5.$$

The first term on the right hand side of this Eq. 19 is the production term for $\tilde{\nu}$ while the second term represents dissipation.

## References

1. Aksoy S, Haralick RM (2000) Feature normalization and likelihood-based similarity measures for image retrieval. Pattern Recognit Lett 22:563–582
2. Amidror I (2002) Scattered data interpolation methods for electronic imaging systems: a survey. J Electron Imaging 11(2):157–176
3. Aranake A, Lakshminarayan V, Duraisamy K (2012) Assessment of transition model and cfd methodology for wind turbine flows. In: 42nd AIAA fluid dynamics conference and exhibit. American Institute of Aeronautics and Astronautics, p 2720
4. Bengio Y (2009) Learning deep architectures for ai. Found Trends Mach Learn 2(1):1–127
5. Carr JC, Beatson RK, Cherrie JB, Mitchell TJ, Fright WR, McCallum BC, Evans TR (2001) Reconstruction and representation of 3d objects with radial basis functions. In: Proceedings of the 28th annual conference on computer graphics and interactive techniques. ACM, New York, NY, USA, SIGGRAPH '01, pp 67–76
6. Chollampatt S, Tou Ng H (2018) A multilayer convolutional encoder–decoder neural network for grammatical error correction. arXiv e-prints arXiv:1801.08831
7. Duraisamy K (2005) Studies in tip vortex formation, evolution and contro, dept of aerospace engineering, Univ of Maryland. PhD thesis, University of Maryland
8. Duraisamy K, Iaccarino G, Xiao H (2019) Turbulence modeling in the age of data. Annu Rev Fluid Mech 51(1):357–377
9. Fernando B, Karaoglu S, Saha SK (2015) Object class detection and classification using multi scale gradient and corner point based shape descriptors. arXiv e-prints arXiv:1505.00432
10. Foley JD, van Dam A, Feiner SK, Hughes JF (1995) Computer graphics: principles and practice in C, 2nd edn. Addison-Wesley, Reading
11. Fuhrmann S, Goesele M (2014) Floating scale surface reconstruction. ACM Trans Graph 33(4):46:1–46:11
12. Guo X, Li W, Iorio F (2016) Convolutional neural networks for steady flow approximation. In: Proceedings of the 22Nd ACM SIGKDD international conference on knowledge discovery and data mining. ACM, KDD '16, pp 481–490
13. Hennigh O (2017) Lat-Net: Compressing Lattice Boltzmann flow simulations using deep neural networks. arXiv e-prints arXiv:1705.09036
14. Hoppe H, DeRose T, Duchamp T, McDonald J, Stuetzle W (1992) Surface reconstruction from unorganized points. SIGGRAPH Comput Graph 26(2):71–78
15. Jürgen S (2015) Deep learning in neural networks: an overview. Neural Netw 61:85–117
16. Kazhdan M, Bolitho M, Hoppe H (2006) Poisson surface reconstruction. In: Polthier K, Sheffer A (eds) Symposium on geometry processing. The Eurographics Association, pp 61–70
17. Koren B (1993) A robust upwind discretization method for advection, diffusion and source terms. Vieweg, Decatur, pp 117–138
18. Lakshminarayan VK, Baeder JD (2010) Computational investigation of microscale coaxial-rotor aerodynamics in hover. J Aircr 47(3):940–955
19. Lecun Y, Bottou L, Bengio Y, Haffner P (1998) Gradient-based learning applied to document recognition. Proc IEEE 86(11):2278–2324
20. Lee S, You D (2017) Prediction of laminar vortex shedding over a cylinder using deep learning. arXiv e-prints arXiv:1712.07854
21. Lee S, You D (2018) Data-driven prediction of unsteady flow fields over a circular cylinder using deep learning. arXiv e-prints arXiv:1804.06076
22. Ling H, Jacobs DW (2007) Shape classification using the inner-distance. IEEE Trans Pattern Anal Mach Intell 29(2):286–299
23. Mathieu M, Couprie C, LeCun Y (2015) Deep multi-scale video prediction beyond mean square error. arXiv e-prints arXiv:1511.05440
24. Medida S, Baeder J (2011) Application of the correlation-based gamma-re theta t transition model to the Spalart–Allmaras turbulence model. American Institute of Aeronautics and Astronautics
25. Miyanawala TP, Jaiman RK (2017) An efficient deep learning technique for the Navier–Stokes equations: application to unsteady wake flow dynamics. arXiv e-prints arXiv:1710.09099
26. Pandya S, Venkateswaran S, Pulliam T (2003) Implementation of preconditioned dual-time procedures in overflow. American Institute of Aeronautics and Astronautics
27. Prantl L, Bonev B, Thuerey N (2017) Generating liquid simulations with deformation-aware neural networks. arXiv e-prints arXiv:1704.07854
28. Pulliam T, Chaussee D (1981) A diagonal form of an implicit approximate-factorization algorithm. J Comput Phys 39(2):347–363
29. Raissi M, Karniadakis GE (2018) Hidden physics models: machine learning of nonlinear partial differential equations. J Comput Phys 357:125–141
30. Raissi M, Yazdani A, Karniadakis GE (2018) Hidden fluid mechanics: a Navier–Stokes informed deep learning framework for assimilating flow visualization data. arXiv e-prints p arXiv:1808.04327
31. Raissi M, Perdikaris P, Karniadakis G (2019) Physics-informed neural networks: a deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 378:686–707

32. Ramachandran P, Zoph B, Le QV (2017) Searching for activation functions. arXiv e-prints arXiv:1710.05941
33. Roe PL (1986) Characteristic-based schemes for the euler equations. Annu Rev Fluid Mech 18(1):337–365
34. Sethian JA (1996) A fast marching level set method for monotonically advancing fronts. Proc Natl Acad Sci USA 93:1591–1595
35. Shijie J, Ping W, Peiyi J, Siping H (2017) Research on data augmentation for image classification based on convolution neural networks. In: 2017 Chinese automation congress (CAC). pp 4165–4170
36. Somers D (1997a) Design and experimental results for the s805 airfoil. Tech. Rep. NREL/SR-440-6917
37. Somers DM (1997b) Design and experimental results for the s809 airfoil. Tech. Rep. NREL/SR-440-6918
38. Somers DM (2004) S814 and s815 airfoils: October 1991–july 1992. Tech. rep
39. Spalart P, Allmaras S (1992) A one-equation turbulence model for aerodynamic flows. American Institute of Aeronautics and Astronautics
40. Taylor GW, Fergus R, LeCun Y, Bregler C (2010) Convolutional learning of spatio-temporal features. Comput Vis ECCV 2010:140–153
41. Tompson J, Schlachter K, Sprechmann P, Perlin K (2016) Accelerating Eulerian fluid simulation with convolutional networks. arXiv e-prints arXiv:1607.03597
42. Turkel E (1999) Preconditioning techniques in computational fluid dynamics. Annu Rev Fluid Mech 31(1):385–416
43. van Leer B (1979) Towards the ultimate conservative difference scheme. v. a second-order sequel to godunov's method. J Comput Phys 32(1):101–136
44. Xu K, Kim VG, Huang Q, Kalogerakis E (2015) Data-driven shape analysis and processing. arXiv e-prints arXiv:1502.06686
45. Zhang D, Lu G (2004) Review of shape representation and description techniques. Pattern Recognit 37(1):1–19
46. Zhang Y, Sung WJ, Mavris D (2017) Application of convolutional neural network to predict airfoil lift coefficient. arXiv e-prints arXiv:1712.10082
47. Zhao H (2005) A fast sweeping method for Eikonal equations. Math Comput 74(250):603–627
48. Zuo Z, Shuai B, Wang G, Liu X, Wang X, Wang B, Chen Y (2015) Convolutional recurrent neural networks: learning spatial dependencies for image representation. In: 2015 IEEE conference on computer vision and pattern recognition workshops (CVPRW). pp 18–26