AIAA SciTech Forum
8-12 January 2024, Orlando, FL
AIAA SCITECH 2024 Forum

10.2514/6.2024-0013

Check for updates

Application of Multi-Fidelity Transfer Learning with Autoencoders for Efficient Construction of Surrogate Models

Yiren Shen*, Harsh C. Patel†, Zan Xu‡, and Juan J. Alonso§
Stanford University, Stanford, CA 94305, USA

The infusion of machine learning techniques into aerodynamic surrogate modeling stands out for its considerable reduction in computational costs, effectively capturing intricate flow dynamics, and complementing costly partial differential equation simulations. Despite their effectiveness, these machine learning-based models grapple with challenges related to generalization and computational demands. Ongoing research endeavors are concentrated on refining their accuracy and efficiency, with a specific emphasis on techniques like transfer learning and the integration of multi-fidelity data. The primary objective of this study is to craft a surrogate model for aerodynamic prediction that strikes a balance between efficiency and accuracy. The strategy involves harnessing the power of transfer learning along with multi-fidelity data, aiming to curtail computational expenses while preserving precision. This is achieved by leveraging abundant low-fidelity data for initial training and subsequently fine-tuning the model with scarce high-fidelity data. The surrogate network, adopting an autoencoder-decoder architecture with super-resolution augmentation, is designed to map low-fidelity flow variables to their high-fidelity counterparts. The model is meticulously trained on a dataset encompassing Euler and Reynolds-averaged Navier-Stokes simulations at three different fidelity levels. The results showcase the model's adeptness in predicting with efficiency and significant computational cost reduction, all while maintaining prediction accuracy at a comparable quality between low-fidelity and high-fidelity computational fluid dynamics results. Although some numerical artifacts and limitations are observed in specific fields and locations, the overall efficiency and accuracy of the model underscore its potential applicability in real-time or multi-query scenarios.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

I. Nomenclature

$\mathcal{A}$	=	aerodynamic residuals
C_D	=	drag coefficient
C_L	=	lift coefficient
C_p	=	pressure coefficient
c	=	chord length
F	=	flux
M	=	Mach number
N_i	=	number of feature layers for a tensor i
$\mathcal{F}^{\text{field}}_{a\to b}$	=	mapping function from fidelity a to b for a field variable
$\mathcal{L}$	=	loss function
q	=	vector of aerodynamic state variables
x_a^f	=	surface mesh coordinates at fidelity f
x_v^f	=	volume mesh coordinates at fidelity f
α	=	angle of attack
ρ	=	density
σ	=	standard deviation of a dataset
ω	=	weights and biases in a neural network

*Ph.D. Candidate, Department of Aeronautics and Astronautics, AIAA Student Member. Equal contribution.
†Ph.D. Candidate, Department of Aeronautics and Astronautics, AIAA Student Member. Equal contribution.
‡Ph.D., Department of Aeronautics and Astronautics, AIAA Student Member.
§Vance D. and Arlene C. Coffman Professor, Department of Aeronautics and Astronautics, AIAA Fellow.

Copyright © 2024 by Yiren Shen, Harsh C. Patel, Zan Xu, Juan J. Alonso. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission.

II. Introduction

A. Motivation

Aerodynamic design necessitates the solution of complex equations encompassing diverse physical phenomena like shockwaves, turbulence, and heat transfer. The complexity of these phenomena underscores the value of computational fluid dynamics (CFD) methods, especially in exploring design solutions for aircraft systems and full configurations. Recent advances in computational techniques, high-performance computing, and efficient algorithms have significantly influenced the role of CFD in aerodynamic design. It plays a crucial part in systematically addressing engineering trade-offs and capitalizing on beneficial interactions among numerous design variables and constraints, which leads to the accelerated generation of improved designs. However, simulating realistic fluid flows for many engineering problems using CFD remains computationally expensive. This is particularly evident in multi-query applications like design optimization and uncertainty quantification, which necessitate solving numerous nonlinear partial differential equations (PDEs) and conducting a substantial number of flow-field evaluations [1–4].

In the realm of aerodynamic design, surrogate models offer a valuable alternative by approximating the full model. They aim to provide faster engineering predictions while maintaining sufficient accuracy in capturing complex relationships between design parameters and aerodynamic performance metrics. Surrogate models provide rapid and cost-effective insights into the design space, expediting the optimization process and enabling quicker, more adaptive design iterations. This becomes particularly compelling when applied across multiple design space explorations or optimizations. The acceleration in design iterations, coupled with enhanced real-time design feasibility, positions surrogate models as essential tools for aerodynamic design applications.

B. Background

The idea of integrating multi-fidelity numerical simulations to construct surrogate models for aerodynamic design has demonstrated effectiveness in reducing overall computational costs [5, 6]. Various methods, including Kriging interpolation, polynomial interpolation, and support vector machines, have been proposed for this purpose [7, 8]. Recently, a growing number of researchers have adopted machine learning (ML)-based tools for surrogate modeling. Data-driven surrogate models, capable of accurately capturing nonlinear aerodynamics, such as flow separation over a wing, transonic flows, or discontinuities from shockwaves, have become a popular approach for replacing costly PDE simulations [9].

In the context of ML models, particularly deep neural network (DNN), their capability to capture complex flow dynamics and provide high-fidelity simulations at a reduced computational cost compared to traditional CFD-based methods has been demonstrated [10–18]. For example, Jiang et al. proposed a context generation network for grid-free spatio-temporal solutions in the Rayleigh-Bénard convection problem [10]. Additionally, Obiols-Sales et al. introduced a convolutional neural network (CNN) for predicting steady-state variables, serving as restart inputs for CFD solvers, thereby accelerating CFD simulations and enhancing the resolution of low-resolution simulation inputs [11, 12].

Another study by Li et al. developed a multi-fidelity aerodynamic model using DNN, integrating experimental and computational data to accurately predict surface pressure distribution on transonic wings, showcasing its superiority over models using solely low-fidelity (LF) or high-fidelity (HF) data [19]. Furthermore, Sun et al. introduced a physics-constrained deep learning approach for cost-effective surrogate modeling in fluid dynamics, employing a structured DNN that integrates Navier–Stokes equations into its loss function, demonstrating high accuracy in simulating internal flows. In summary, ML-based surrogates exhibit superior performance over traditional methods and prove to be robust and scalable when appropriately designed and trained [18].

Despite the asserted benefits of ML-based surrogate modeling, several challenges persist, impeding the broader adoption of such tools. The intricacy of these models demands substantial computational resources for gathering extensive training data and for the training process itself [10, 20]. Furthermore, the proposed models often tend to be specific to particular cases, and their ability to generalize across diverse flow conditions and geometries can still be limited [12]. Consequently, the pursuit of a relatively inexpensive surrogate model, capable of reducing the overall computational cost in multi-query aerodynamic design workflows without a significant sacrifice in prediction accuracy, remains an active research frontier.

An effective strategy for mitigating the training cost associated with building a ML-based surrogate model involves employing transfer learning on a multi-fidelity dataset. This entails using a dataset that encompasses multiple fidelity levels, where generating a LF dataset is cost-effective. The initial training of the surrogate model leverages this LF dataset. Subsequently, the LF surrogate model undergoes fine-tuning on a HF dataset through so-called transfer

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

learning (TL), leading to the creation of application-specific, HF surrogate models.

Several prior studies have successfully adopted this approach, showcasing computational cost savings without significant compromises in prediction accuracy [12, 15, 21]. For instance, Song and Tartakovsky demonstrated the application of a multi-fidelity modeling approach in solving surrogate-based multi-phase flow problems, utilizing a dataset comprising a large number of coarse mesh (LF) solutions and a comparatively smaller set of fine mesh (HF) solutions [21]. The methodology proposed by Obiols-Sales et al. also incorporates different levels of spatial resolution to expedite surrogate model training [12]. Additionally, Kou et al. employed a TL network based on domain adaptation using transfer component analysis for constructing reduced-order models in multi-fidelity flow reconstruction [15, 22]. Introducing a multi-fidelity CNN surrogate model with TL for aerodynamic shape optimization, Liao et al. applied it to optimize NACA0012 and RAE2822 airfoils, demonstrating a substantial reduction in computational cost and enhanced optimization efficiency [23].

C. Overview

Motivated by these research advancements, this study aims to develop a surrogate model utilizing a TL framework based on multi-fidelity data. These data encompass varying levels of physical modeling complexity and spatial resolution, specifically tailored for a designated aerodynamic design exploration application. The objective is to capitalize on the abundance of LF data for training and employ TL to adapt the model for HF, consequently reducing the overall computational cost compared to HF CFD simulations. At the core of our approach is the efficient construction of an accurate surrogate model that not only upholds the requisite precision for complex aerodynamic analyses but also significantly diminishes the demand for computational resources. This provides a robust and versatile tool for advanced aerodynamic design and optimization.

III. Methods

A. Governing Equations

The Navier-Stokes equations describe the behavior of Newtonian fluids under the continuum hypothesis assumption. The finite-volume flow solver in SU2 allows the discretization and solution of the compressible Navier-Stokes equations, which can be expressed in differential form as

$$\mathcal{A}\left(q,\, \nabla q\right) = \frac{\partial q}{\partial t} + \nabla \cdot F_c\left(q\right) - \nabla \cdot F_v\left(q,\, \nabla q\right) - S\left(q\right) = 0\,, \tag{1}$$

where $q = [\rho,\, \rho u,\, \rho v,\, \rho e]^T$ represents the vector of flow state variables in 2D at a given cell center. In this notation, ρ is the fluid density, u and v are the fluid velocity components in the x and y directions, and e is the total energy per unit mass. The convective fluxes are denoted by $F_c\left(q\right)$, the viscous fluxes by $F_v\left(q,\, \nabla q\right)$, and $S\left(q\right)$ represents a generic source term. It's essential to note that the inviscid Euler equations can be obtained by excluding the viscous fluxes $F_v\left(q,\, \nabla q\right)$ in (1).

The Reynolds-averaged Navier-Stokes (RANS) equations provide a cost-effective approximation to the full Navier-Stokes equations, enabling feasible CFD simulations for flows with high Reynolds numbers and complicated geometries. These equations result from assuming that the flow comprises a mean velocity flow-field with perturbations. The mean time-averaged value and the time interval of the averaging are chosen to be large enough concerning the time scales of the turbulent fluctuations but small enough to capture other large-scale time-dependent flow features. One-equation models involve introducing an additional transport equation to compute the turbulence kinematic viscosity. In these experiments, Spalart-Allmaras (SA) model [24], specifically designed and optimized for flows past wings and airfoils, demonstrates excellent performance and is utilized.

B. Aerodynamic Dataset

The SU2 [25–27] CFD solver is utilized to simulate the aerodynamics of a 2D NACA0012 airfoil, employing both inviscid Euler and viscous RANS fluid models to generate the dataset for surrogate model training and validation. The flow domain is discretized using a C-type mesh, and simulations are conducted for the airfoil at angles of attack α_∞ ranging from 0 to 5.0° in increments of 0.1° relative to the direction of the freestream flow. Additionally, the freestream Mach number M_∞ is varied across 0.3, 0.5, and 0.7. This variable sweep over the freestream angle of attack and Mach

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

number results in a total of 153 samples each for low-fidelity (LF), medium-fidelity (MF), and high-fidelity (HF) models, all converging to 10^{-10}.

The dataset encompasses flow field at three fidelity levels: a coarse grid Euler simulation (q^{LF}), a coarse grid RANS simulation (q^{MF}), and a fine grid RANS simulation (q^{HF}). A summary of solver and mesh statistics for each fidelity level is provided in Table 1. The mesh employed for q^{LF} is identical to that used in q^{MF}, while q^{HF} utilizes a substantially finer mesh. Figure 1 illustrates the differences in the dataset across fidelities, presenting the Mach number contour at $\alpha_\infty = 5^\circ$ and $M_\infty = 0.7$, with the mesh overlaid in faint white color. Subsequently, the generated dataset undergoes random partitioning into training and validation sets, with a 70 %–p30 % split based on freestream conditions. This partition is depicted in Figure 2, with freestream conditions selected for the training set colored in blue and those for the validation set colored in gray.

The dataset is generated on the Sherlock cluster at the Stanford Research Computing Center (SRCC), utilizing a 32-core 2.4 GHz partition. The total core-hours for generating q^{LF}, q^{MF}, and q^{HF} datasets are 4.7 h, 5.1 h, and 161.7 h, respectively. The averaged core-hours required for computing one run are provided in the last column of Table 1.

Model Type	Mesh Type	Fluid Model	Volume Elements	Surface Elements	Avg. Time
Low-fidelity (LF)	Coarse	Euler	9900	77	0.031
Medium-fidelity (MF)	Coarse	RANS	9900	77	0.033
High-fidelity (HF)	Fine	RANS	163 785	317	1.057

Table 1 Summary of different levels of dataset fidelity with key statistics of mesh sizes and average computation time reported in core-hours.

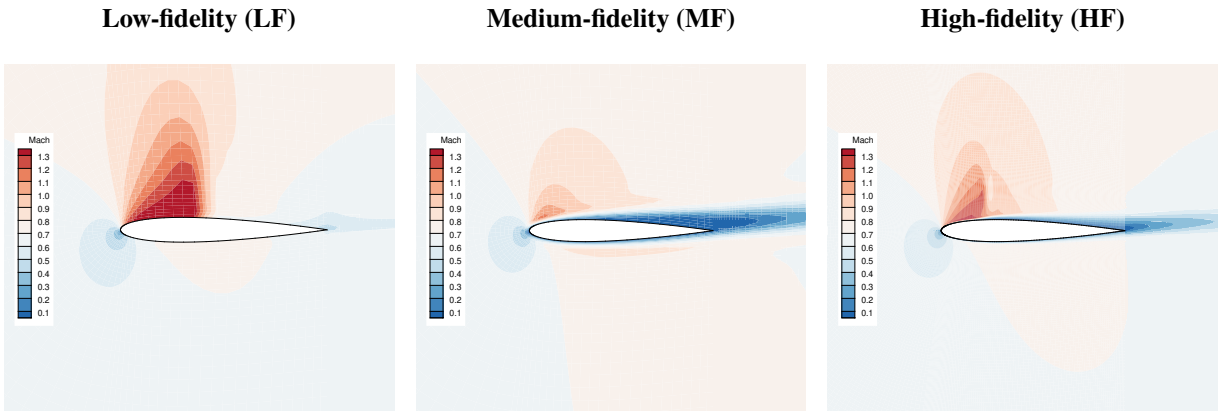


Fig. 1 The 2D computational domain for the NACA0012 and the corresponding Mach number contours for LF, MF, and HF simulations.

C. Surrogate Modeling

The main objective of this ML-based surrogate model is to establish a mapping function $\mathcal{F}$ that transforms LF flow variables q^{LF} into their HF counterparts q^{HF}. This mapping is expressed as

$$q^{\mathrm{HF}} \sim \mathcal{F}\left(q^{\mathrm{LF}}, \omega_{\mathcal{L}}\right), \tag{2}$$

where $\omega_{\mathcal{L}}$ represent the weights and biases of the network under a specific loss function $\mathcal{L}$. To establish the mapping of flow variables across these three fidelities, two mapping functions, $\mathcal{F}_{\mathrm{LF}\to\mathrm{MF}}$ and $\mathcal{F}_{\mathrm{MF}\to\mathrm{HF}}$, are necessary. Two ML-based blocks are employed to reconcile these mappings. The following sections provide a detailed discussion on the architectures and the training pipeline.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

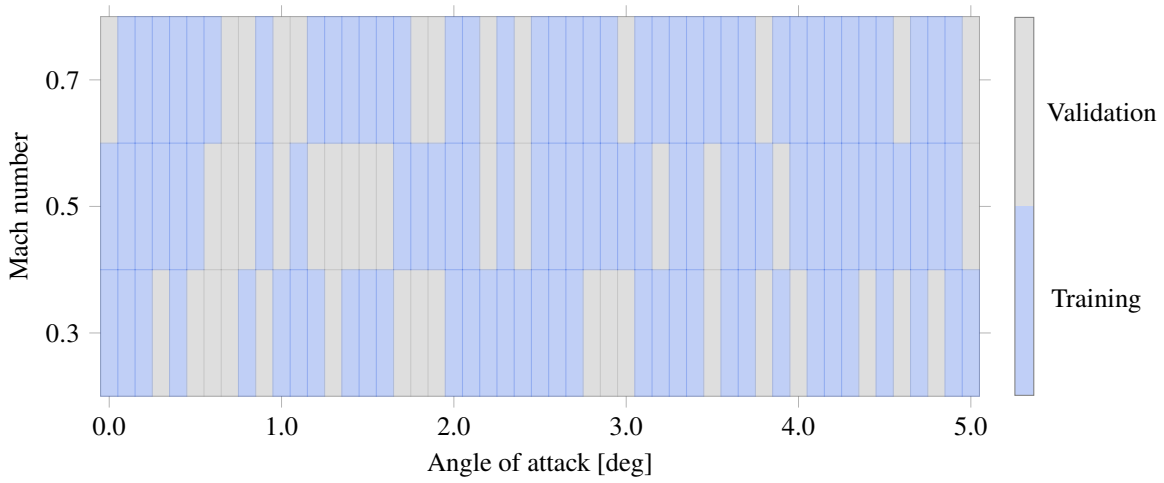


Fig. 2 Overview of the dataset, with blue color filling indicating test runs for training, while gray is used for test runs designated for validation.

1. Mapping from Low-Fidelity to Medium-Fidelity (`ER Block`)

The flow fields of the Euler simulation q^{LF} and the coarse grid RANS simulation q^{MF} exhibit distinct differences stemming from the omission of viscous fluxes, as explained in Section III.A. The mapping from q^{LF} to q^{MF} thus necessitates a feature extraction reconstruction network capable of mapping between different fidelities of physical modeling. The proposed network takes the form of an autoencoder-decoder network, drawing heavy inspiration from the work of Patil et al. on emulating turbulent viscosities [20]. For the mapping function $\mathcal{F}_{\mathrm{LF}\to\mathrm{MF}}$, a similar autoencoder network is implemented and referred to as the Euler-RANS block (`ER Block`).

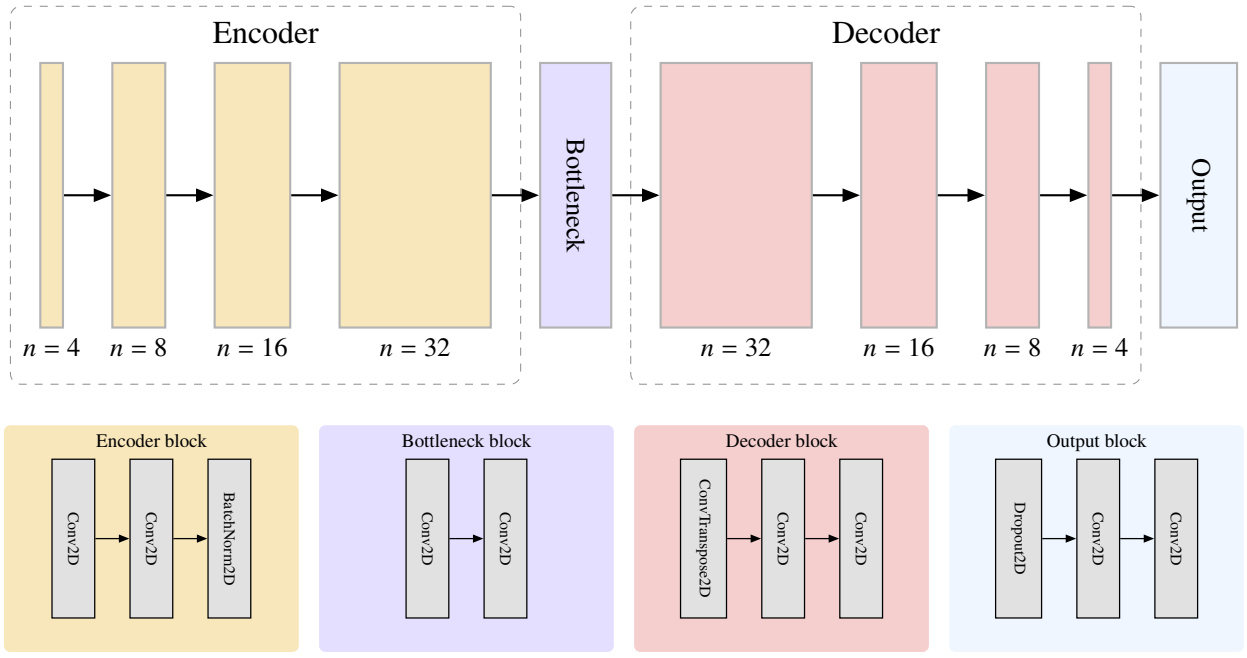


Fig. 3 The `ER Block` architecture is designed to map coarse grid Euler flow fields to coarse grid RANS flow fields. Each functional block is denoted by a distinct color: encoder (yellow), bottleneck (purple), decoder (red), and output (blue). The corresponding block depth is annotated at the bottom.

The `ER Block`, depicted in Figure 3, comprises four main components arranged sequentially: the encoder (yellow),

the bottleneck (purple), the decoder (red), and the output blocks (blue). Within the model, the depth of the feature layer progressively increases from N_{input} to $2N_{\text{input}}$, $4N_{\text{input}}$, and $8N_{\text{input}}$, facilitated by the four encoder blocks before reaching $N_{\text{bottleneck}}$ in the bottleneck block. The bottleneck block consists of two convolutional layers with a kernel size of 3 and a stride of 1, maintaining the feature depth. Subsequently, the four decoder blocks sequentially decrease the feature depth from $N_{\text{bottleneck}}$ to $8N_{\text{input}}$, $4N_{\text{input}}$, $2N_{\text{input}}$, and finally to N_{input}. The final output block, consisting of two convolutional layers, reduces the feature depth to the desired output field depth (N_{output}).

In this design, each encoder block comprises two convolutional layers followed by one batch normalization layer and concludes with an activation function layer. The convolutional layers are structured to maintain consistent input and output channel depths, with the second convolutional layer having a higher stride ($\text{stride}_{\text{conv1}} = 1$, $\text{stride}_{\text{conv2}} = 2$) than the first. This design enables the network to progressively capture more complex features, halve the spatial dimensions of the tensor after each encoder block, and perform offset scanning over the entire fluid domain. A Leaky ReLU activation function is employed at the end of each encoder block for model stability.

The decoder blocks, mirroring the encoder blocks but in reverse order, gradually upsample and reduce the channel depth to reconstruct or transform the input. Each decoder block includes one transposed convolutional layer and two convolutional layers. The transposed convolutional layer doubles the spatial dimension of the tensor while halving the feature depth. The subsequent convolutional layers maintain the spatial dimensions and feature depth of the tensor. Notably, there are no activation functions embedded in the decoder blocks.

For all convolutional layers, the weights ω are initialized using the Xavier initialization method [28], which improves training stability. In total, the network comprises 353 796 trainable parameters.

The input to this network comprises flow variables q^{LF} computed at a coarse grid using the Euler solver. Standard z-score normalization [29], utilizing statistics computed from the entire dataset, is applied to normalize the input flow field data. The original CFD simulation involves a C-shaped structured grid. A flattening scheme is employed to convert the C-shaped domain along the airfoil surface into a rectangular domain, utilizing clockwise nodal connectivities. After flattening, special attention is given to boundary conditions, particularly at the edge comprising the airfoil boundary and the extension of the trailing edge to the far-field. Following the approach in previous studies [20], padding with halo cells is implemented, which ensures that periodic boundary conditions are padded with appropriate values from the periodic cells, and wall boundary conditions are enforced by padding zeros in the corresponding halo cells. The network's output is flow field emulating q^{MF} computed from a coarse grid RANS solution.

The biases and weights $\omega_{\mathcal{L}}$ are optimized by minimizing the loss function ($\mathcal{L}$) which is defined as

$$\mathcal{L} = \frac{1}{n}\sum_i \left(q^{\text{HF}} - \mathcal{F}\left(q^{\text{LF}}, \omega_{\mathcal{L}}\right)\right)^2 + \lambda\sqrt{\frac{1}{n}\sum_i (q_b^{\text{HF}} - \mathcal{F}(q_b^{\text{LF}}, \omega_{\mathcal{L}}))^2}\,, \tag{3}$$

where n is the batch size and the loss function is composed of two terms: the first term represents the mean-square error (MSE) of predictions compared to targets in a training batch, and the second term is the root-mean-square error (RMSE) of predictions at the boundary cells of the airfoil q_b^{HF} relative to targets. The inclusion of the second term enforces a stricter constraint on model predictions at the airfoil surface, especially when $\lambda > 1$. In the current study, $\lambda = 5$ is chosen, striking a good balance between training convergence and the quality of predictions.

2. Mapping from Medium-Fidelity to High-Fidelity (`SR Block`)

Numerous prior investigations have focused on developing surrogate models that map low spatial resolution to high spatial resolution within multi-fidelity dataset inputs [12, 21, 30]. With the ML-based model proposed in the preceding section, a minor adjustment in the network architecture — specifically, in the convolutional kernel size of the decoder blocks — can enable such mapping. However, constructing these models typically requires a substantial training dataset due to the network complexity, and tweaking the architecture of the `ER Block`'s decoder results in changes to multiple decoder weights ω. Given that generating q^{HF} is significantly more resource-intensive than q^{LF}, and considering the aim of developing an efficient, compact surrogate model to expedite design exploration, accumulating a large dataset of $\mathcal{F}_{\text{LF}\to\text{HF}}$ pairs is impractical. Therefore, we explore methods that are compact enough, resulting in a small size of ω, to be trained with a limited number of q^{HF}.

In computer vision, the challenge of mapping between different spatial resolutions is known as the so-called *super-resolution* problem. Notably, there has been a recent surge in proposals for networks that exhibit compactness, featuring on the order of tens of thousands of trainable parameters, making them amenable to training with datasets ranging from tens to hundreds of LF-HF pairs [31–34]. Following an assessment of the prediction accuracy across a set of 33 training images, the sub-pixel convolutional neural network (SPCNN) network, introduced by Shi et al.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

[33], emerged as a favorable choice for these investigations. This decision is grounded in the inherent compactness of the SPCNN architecture, demonstrating superior super-resolution results when trained with 33 $(q^{\mathrm{MF}}, q^{\mathrm{HF}})$ pairs. Consequently, employing SPCNN as the final output block in the network proposed in `ER Block` aligns seamlessly with our objectives, aiming to achieve optimal super-resolution predictions while preserving the overall efficiency and robustness of the surrogate network.

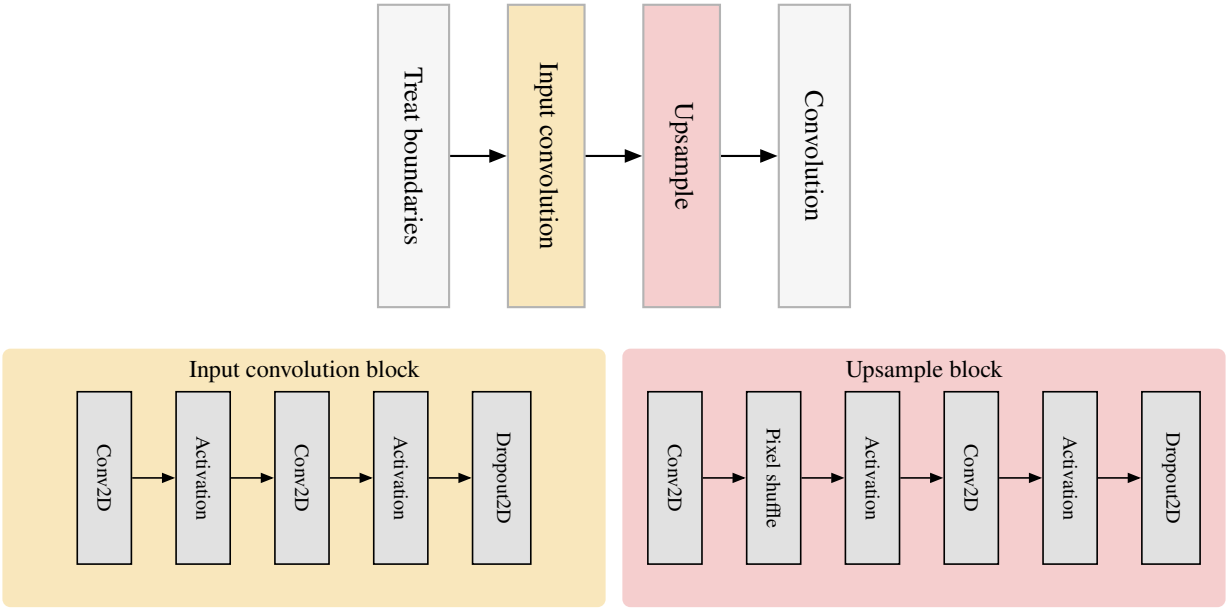


Fig. 4 Architecture of the `SR Block`, with each functional block represented by a distinct color: input convolution (yellow), and upsample (red).

As depicted in Figure 4, the super-resolution block (`SR Block`) consists of three main blocks: the input convolution block (highlighted in yellow), the sub-pixel upsampling block (colored in red), and the output convolutional layer. The input convolution block comprises two 2D convolutional layers, each followed by a hyperbolic tangent activation function. A dropout regularization layer with a dropout rate of 0.1 is integrated into the input convolution block, and this dropout rate was determined to be optimal through hyperparameter tuning. The sub-pixel upsampling block, sharing a similar architecture with the input convolution block, incorporates a pixel shuffle operation after the first convolutional layer. The pixel shuffle layer rearranges values pixel-wise along the feature depth direction and redistributes them across the spatial dimension [33]. The final convolutional layer maps the tensor to the desired output feature depth N_{output}, set to $N_{\mathrm{output}} = 2$ for momentum and $N_{\mathrm{output}} = 1$ for density and energy. In this network, the feature depth increases from N_{input} to $16, 64, 128$ just before the pixel shuffle layer, and then decreases from $128, 8$ to N_{output} after the last convolutional layer.

During experimentation, noticeable numerical artifacts were observed near the boundary in the network output. To address these artifacts at the far-field boundaries and enhance prediction accuracy at the airfoil surface, the input is padded with two layers of halo cells, mirroring the boundary cell values. For the periodic boundary condition, extending from the trailing edge to the far-field edge, halo cells are appropriately padded with corresponding values from their periodic counterparts at the boundary.

3. Transfer Learning

After training the `ER Block` on a $(q^{\mathrm{LF}}, q^{\mathrm{MF}})$ and the `SR Block` on an augmented smaller set of $(q^{\mathrm{MF}}, q^{\mathrm{HF}})$ pairs, these two networks are concatenated to create a surrogate model $\mathcal{F}_{\mathrm{LF}\to\mathrm{HF}}$. This model is designed to map q^{LF} to their q^{HF} counterparts. The first mapping from q^{LF} to q^{MF} by the `ER Block` is denoted as $\mathcal{F}_{\mathrm{LF}\to\mathrm{MF}}$, and the second mapping from q^{MF} to q^{HF} by the `SR Block` is referred to as $\mathcal{F}_{\mathrm{MF}\to\mathrm{HF}}$. Ideally, the composition of these two mappings should reconstruct q^{HF} from q^{LF} with minimal loss. However, due to the likelihood of prediction errors and numerical artifacts in the predicted $q^{\mathrm{MF}}_{\mathrm{pred}}$, a direct composition of the two mappings often results in suboptimal prediction accuracy.

To address the aforementioned challenges, the study employs TL throughout the training process. Transfer learning

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

is a ML technique that involves utilizing model weights trained for a specific task as the starting point for another model on a related task. This approach leverages the knowledge acquired during the solution of the first task, usually trained with abundant data, to expedite and enhance model learning for a related but distinct task [35].

In this application, the TL approach integrates the `SR Block` with the `ER Block`. This TL strategy eliminates the need for generating a computationally demanding dataset to train a surrogate model directly on HF data. As illustrated in Figure 5, input and output variables are depicted in the gray boxes, while the `ER Block` and `SR Block` are placed in yellow and red boxes, respectively. In Phase 1, the `ER Block` is trained with $\left(q^{\mathrm{LF}}, q^{\mathrm{MF}}\right)$ generating intermediate $q^{\mathrm{MF}}_{\mathrm{pred}}$ output. A `SR Block` that is pre-trained, with the training sequence not shown in Figure 5, is subsequently concatenated to the `ER Block`. The `SR Block` with pre-trained weights is denoted by the decoders with dashed edges in Phase 2. Following this, in Phase 2, fine-tuning training is conducted on the `SR Block` taking $q^{\mathrm{MF}}_{\mathrm{pred}}$ as input and q^{HF} as output target, while the trained weights for the `ER Block` remain unmodified. The pre-trained `SR Block` weights are allowed to fine-tune and adapt to reject the noise and artifacts of the emulated $q^{\mathrm{MF}}_{\mathrm{pred}}$ mentioned earlier. This is achieved using a small gradient descent step size, and typically, 10 to 30 training epochs suffice for this final phase of model tuning.

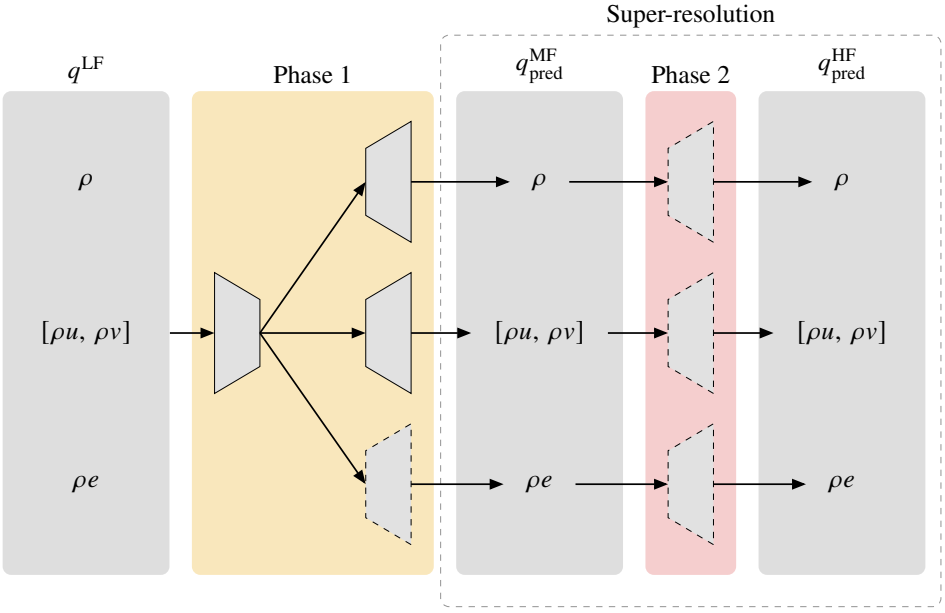


Fig. 5 Transfer learning workflow used in the current study. Input and output for various flow field fidelities are listed in gray boxes. Phase 1 training of `ER Block` is indicated in yellow, and Phase 2 training of `SR Block` is in red. A converging trapezoid (left to right) represents an autoencoder and a diverging trapezoid represents a decoder. Decoders initialized with pre-trained weights are marked with dashed edges.

The incorporation of TL expedites the convergence of training when working with the `ER Block`. In Phase 1, as illustrated in Figure 5, the decoder associated with ρe is denoted with a dashed edge, indicating that its weights ω are initialized using TL. Since the decoder for ρe is trained last in the training sequence, we initialize its weights using the trained weights from the decoder for ρ. This approach results in a significant reduction in training time, as highlighted in Section IV.A.

IV. Results

A. Network Training and Hyperparameters Optimization

The architecture detailed in Section III is implemented using PyTorch [36]. The initial training involves the `ER Block` as part of the network outlined in Phase 1 in Figure 5. This training takes place on a NVIDIA A100 graphics processing unit (GPU) with a batch size of 64, utilizing the ADAM optimizer [37] with an initial learning rate of 8×10^{-3}. A learning rate decay scheduler decreases the learning rate by 20 % every 500 epochs. The training process for `ER Block`, responsible for mapping from $\mathcal{F}_{\mathrm{LF}\rightarrow\mathrm{MF}}$, is divided into three sub-phases.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

In the first sub-phase, the autoencoder-decoder for the density field ($\mathcal{F}^{\rho}$) is trained over 5000 epochs. The z-score normalization is applied to the network's input q^{LF} using mean and variance values from the complete dataset of 153 test runs, as explained in Section III.C. To enhance the model's robustness, random noise with a magnitude of 3 % of the standard deviation σ is added to q^{LF} at each epoch. The validation compound loss decreases from 1089.3 to 0.14, indicating convergence.

In the second sub-phase, the autoencoder-decoder for the momentum field undergoes training for 5000 epochs under conditions similar to those in training $\mathcal{F}^{\rho}$, with the exception that the encoder weights, transferred from $\mathcal{F}^{\rho}$, remain frozen. The validation loss decreases from 99.7 to 1.39 over these epochs.

In the third sub-phase, the decoder for the energy field is trained, initializing with weights from both the encoder and decoder from $\mathcal{F}^{\rho}$ while keeping the encoder weights frozen. Due to these transferred weights, only 3000 epochs are required to train the energy decoder. The validation loss decreases from 5.41 to 0.10. The total training time for $\mathcal{F}_{\mathrm{LF}\rightarrow\mathrm{MF}}$ across all fields is 15 min.

The transformation from $\mathcal{F}_{\mathrm{MF}\rightarrow\mathrm{HF}}$ is executed by the `SR Block`, detailed in Section III.C.2. To showcase an efficient construction of a surrogate model while considering the high cost of generating q^{HF}, only 44 $\left(q^{\mathrm{MF}},\ q^{\mathrm{HF}}\right)$ data pairs are utilized for training and validation. These pairs, designated as "validation" data in Figure 2, are randomly split into training and validation sets in a 75 %-25 % ratio. In addition to the boundary treatment of the q^{MF} discussed in Section III.C.2, the training input undergoes further augmentation by random cropping using a window at half the size of the input tensor dimension to enrich training samples. A batch size of 32 is selected for training, with an initial learning rate of 10^{-3}. A learning rate decay scheduler reduces the rate by 20 % every 500 epochs, and the compound loss in (3) is employed for validation set evaluation.

To enhance super-resolution robustness, random noise with a magnitude of 7 % of the standard deviation is added to the input q^{MF} at each training epoch. The q^{MF} input dataset undergoes z-score normalization. For the momentum field, training continues for 5000 epochs or until no significant further reduction in validation loss is observed. The validation compound loss decreases from 696.5 to 7.8. For the energy and density fields, the same mapping function is used due to the similar data structures of these flow variables. Training proceeds for 5000 epochs or until no further significant validation loss reduction is observed. The validation compound loss drops from 2.3 to 0.8. The total corresponding training time is 7 minutes.

In the concluding training phase, denoted as Phase 2 in Figure 5, the `ER Block` is combined with the `SR Block`. In this phase, the weights of the trained `ER Block` are kept fixed, and the `SR Block` is initialized with trained weights. The primary objective of this phase is to train the `SR Block` to adapt to the predicted $q^{\mathrm{MF}}_{\mathrm{pred}}$ from Phase 1, which may contain numerical artifacts and noise unlike the q^{MF} inputs into `SR Block` during its pre-training. The network undergoes training on the same set of test runs used for the `SR Block` pre-training, but with q^{LF} as the input. The output predictions $q^{\mathrm{HF}}_{\mathrm{pred}}$ are then compared against q^{HF}. A random 75 % - 25 % dataset split again divides the test runs into training and validation sets. The compound loss function in (3) is applied for this phase of the training as well.

The training employs a batch size of 32, a learning rate of 10^{-8}, and spans over 10 epochs, completing within 1 min. This duration is chosen based on the observation that further training results in oscillatory validation loss values without a significant reduction. Considering that the training set for Phase 2 is limited, prolonged training may lead to overfitting on the small validation set, which includes only 11 test runs. The surrogate model's training concludes after Phase 2.

A preliminary investigation of model hyperparameters is conducted to determine optimal values based on subsequent evaluation metrics. The chosen activation function is Leaky ReLU with an activation slope set to 8×10^{-2}. To identify the optimal slope, a hyperparameter sweep study is undertaken, exploring values within the range 0.005, 0.008, 0.01, 0.02. The activation slope significantly impacts feature extraction in the `ER Block` encoder; an excessively small or large slope hinders the encoder's ability to capture specific flow features, such as the wake, or results in excessive sensitivity to the injected noise. The selected slope demonstrates low validation loss while accurately reconstructing the flow field, as assessed through manual judgment.

Another hyperparameter sweep focuses on the loss function weight, denoted as λ in (3), with values explored in 0.1, 1, 3, 5, 10. Model performance is evaluated based on the error between calculated surface C_L and C_D and their ground truth values, as well as the quality of the reconstructed RANS-like flow field, assessed through manual judgment. Increasing λ reduces the aerodynamic coefficient prediction error while elevating the MSE error of the flow field on the validation set. The chosen value of $\lambda = 5$ strikes a balance between accurate aerodynamic coefficient predictions and acceptable flow field predictions.

The final hyperparameter examined is the bottleneck size, denoted as $N_{\mathrm{bottleneck}}$, with values considered in 32, 64, 96. An increase in $N_{\mathrm{bottleneck}}$ generally enhances prediction accuracy for the validation set but also escalates model complexity and training hardware requirements. The ultimate model adopts $N_{\mathrm{bottleneck}} = 96$ to achieve optimal performance within

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

the constraints of model complexity.

B. Aerodynamic Predictions

The main objective of this paper is to develop an efficient surrogate model suitable for multi-query applications, striving for prediction accuracy comparable to that of HF CFD simulations. To comprehensively assess the model's predictive performance, we extend our analysis beyond the proposed loss function outlined in (3). This expanded evaluation encompasses two distinct components. Firstly, we perform a quantitative analysis of the surface energy on the airfoil, enabling the computation of lift coefficient C_L and drag coefficient C_D. Secondly, we undertake a qualitative assessment by manually comparing the differences between $q^{\text{HF}}_{\text{pred}}$ and q^{HF}.

Our analysis commences by evaluating the MF prediction capabilities of the `ER Block` in its standalone form, without integrating the super-resolution block. This involves assessing the performance of $\mathcal{F}_{\text{LF}\to\text{MF}}$ against the q^{MF} from CFD results, which serve as the groundtruth. We hypothesize that $\mathcal{F}_{\text{LF}\to\text{MF}}$ can learn features such as boundary values, pressure distribution, and momentum distribution relative to the field boundaries from the q^{LF} input and can reconstruct a RANS-like solution at a spatial resolution similar to that of MF. Various metrics are available for the quantitative comparison of prediction quality, including the MSE calculated across the entire flow field or within a specific region of interest. We choose to compare the surface total energy on the airfoil, which is particularly relevant for applications where the surrogate model is required to predict airfoil C_L and C_D. These quantities are derived from the surface pressure, which, in turn, can be deduced from the total energy at the airfoil surface.

By defining the dynamic pressure $q = \frac{1}{2}\rho V^2$, where ρ is the density, V is the velocity, and taking the specific heat ratio γ, the surface pressures can be determined by

$$p = (\gamma - 1)(\rho e - q) \tag{4}$$

and the corresponding pressure coefficient

$$C_p = \frac{p - p_\infty}{q_\infty} \tag{5}$$

based on the freestream pressure p_∞ and freestream dynamic pressure q_∞. Subsequently, the corresponding lift coefficient C_L and drag coefficient C_D are calculated as

$$C_L = \int_s C_p \sin(\theta) ds \tag{6}$$

$$C_D = \int_s C_p \cos(\theta) ds\,, \tag{7}$$

where s represents the airfoil surface, and θ denotes the local surface inclination. The computation of C_L and C_D involves integrating the expressions over the mesh grid using numerical integration.

To examine the predicted surface energy, we assess the surface energy outputs at $M_\infty \in \{0.3, 0.5, 0.7\}$ and $\alpha_\infty \in \{0.0, 1.0, 2.0, 3.0, 4.0, 5.0\}^\circ$. The input for this analysis is q^{LF}, provided without artificially introduced noise but with boundaries padded. The mapping $\mathcal{F}_{\text{LF}\to\text{MF}}$ represents the trained model of the `ER Block`.

To quantify the relative error between the prediction and the ground truth, we define a normalized surface energy error as $(e_t - e_p)/e_\infty$, where e_t is the target surface energy computed using CFD at MF, and e_p is the predicted surface energy with e_p from $\mathcal{F}_{\text{LF}\to\text{MF}}$. The normalized surface energy errors for selected test runs are presented in Figure 6, arranged with increasing Mach number from left to right and increasing angle of attack from top to bottom. In each figure, the x-axis represents the surface coordinate normalized by camber x/c, and the y-axis denotes the normalized surface energy error. In these plots, the blue line indicates the error on the upper surface and the red line shows the error on the lower surface.

As illustrated in Figure 6, the highest errors typically occur at the leading edge of the airfoil ($x/c = 0$) and diminish toward the trailing edge ($x/c = 1$). Within each test, predictions on the lower surface generally exhibit greater accuracy than those on the upper surface. This difference is speculated to stem from the more intricate flow dynamics on the upper surface. Across the test cases, an increase in error is noticeable with higher values of α_∞ and M_∞. The underlying cause of this trend is presently under investigation.

Notably, at $M_\infty = 0.7$, a shockwave is observed on the upper surface of the airfoil. The presence of this shock, combined with the observation that the location of maximum error for the test case at $\alpha_\infty = 5^\circ$ and $M_\infty = 0.7$ coincides with the shock location in CFD results, suggests that the trained network may encounter challenges in accurately

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

reconstructing RANS-like results in regions with flow nonlinearities. However, it is crucial to acknowledge that the distribution of validation runs is randomized. This observed effect might be an artifact of the training process due to an uneven distribution of validation runs toward lower α_∞ and M_∞ depicted in Figure 2. To delve deeper into this phenomenon, further investigation involving training with multiple randomized datasets is necessary to eliminate potential training artifacts, and this will be a focus of future work. Nevertheless, for the majority of runs, the normalized surface energy error remains below 1 % across the airfoil surface, indicating that the trained model $\mathcal{F}_{\text{LF}\to\text{MF}}$ is adequately learns RANS-like flow fields from Euler simulation inputs.

To demonstrate the efficacy of the RANS-like HF prediction method, we present a comparative test case evaluating the flow field and absolute error. Figure 8 illustrates this comparison, where the left column showcases the HF predictions $q^{\text{HF}}_{\text{pred}}$, the center column depicts the target flow variables q^{HF}, and the right column plots the relative errors computed as

$$\text{error} = 100 \times \left| \frac{q^{\text{HF}} - q^{\text{HF}}_{\text{pred}}}{q^{\text{HF}}} \right|, \tag{8}$$

where the variables ρ, ρu, ρv, and ρe are displayed from top to bottom. The predicted results exhibit a RANS-like boundary layer around the airfoil and a viscous wake, particularly evident in the ρu plot in the second row. It is important to emphasize that these viscous features are absent in the LF input. However, numerical artifacts such as spurious density variations in the far-field and concentric circular patterns around the leading edge of the airfoil in the x- and y-momentum are also present. Other deviations include a thinner boundary layer around the airfoil and faster momentum dissipation in the wake region. The observed high relative error in the y-momentum primarily results from divisions by small magnitudes, especially since the y-momentum approaches zero in the far field.

Despite these limitations, it is essential to compare computational costs across different methods. Specifically, the LF Euler CFD approach, executed on a 2.4 GHz central processing unit (CPU), requires 3.08×10^{-2} core-hours, and the HF RANS CFD method demands 3.35×10^{-1} core-hours. Remarkably, the proposed surrogate model demonstrates a significant reduction in computation time, requiring only 8.33×10^{-7} core-hours, thereby highlighting its computational efficiency.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

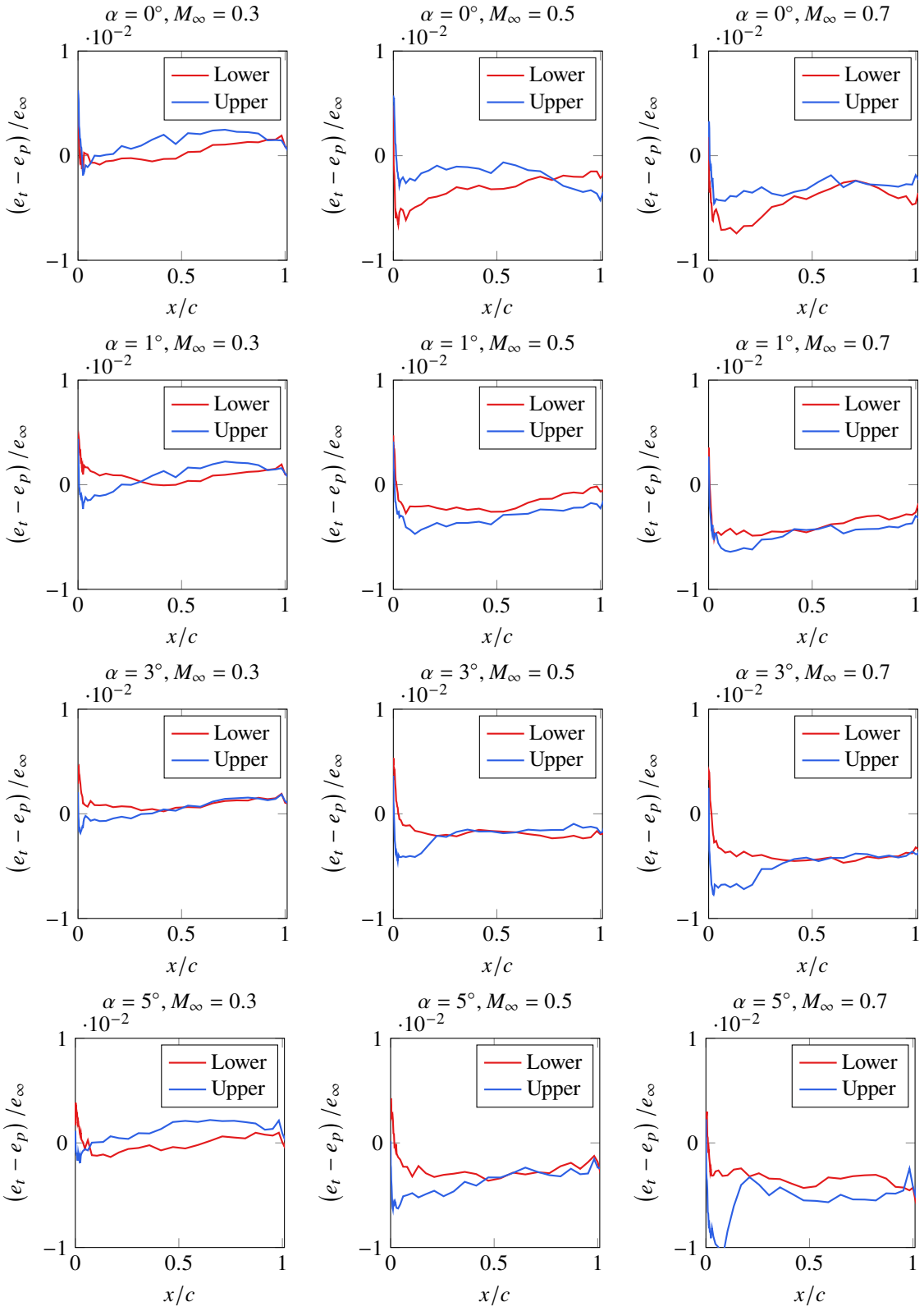


Fig. 6 Comparison of the normalized surface energy error between predictions of `ER Block` and MF CFD. The normalized error is plotted against the normalized coordinate (x/c) from the leading edge to the trailing edge of the airfoil. Results are organized by test runs, with M_∞ of $0.3, 0.5, 0.7$ (from left to right) and α_∞ of $0°, 1°, 3°, 5°$ (from top to bottom).

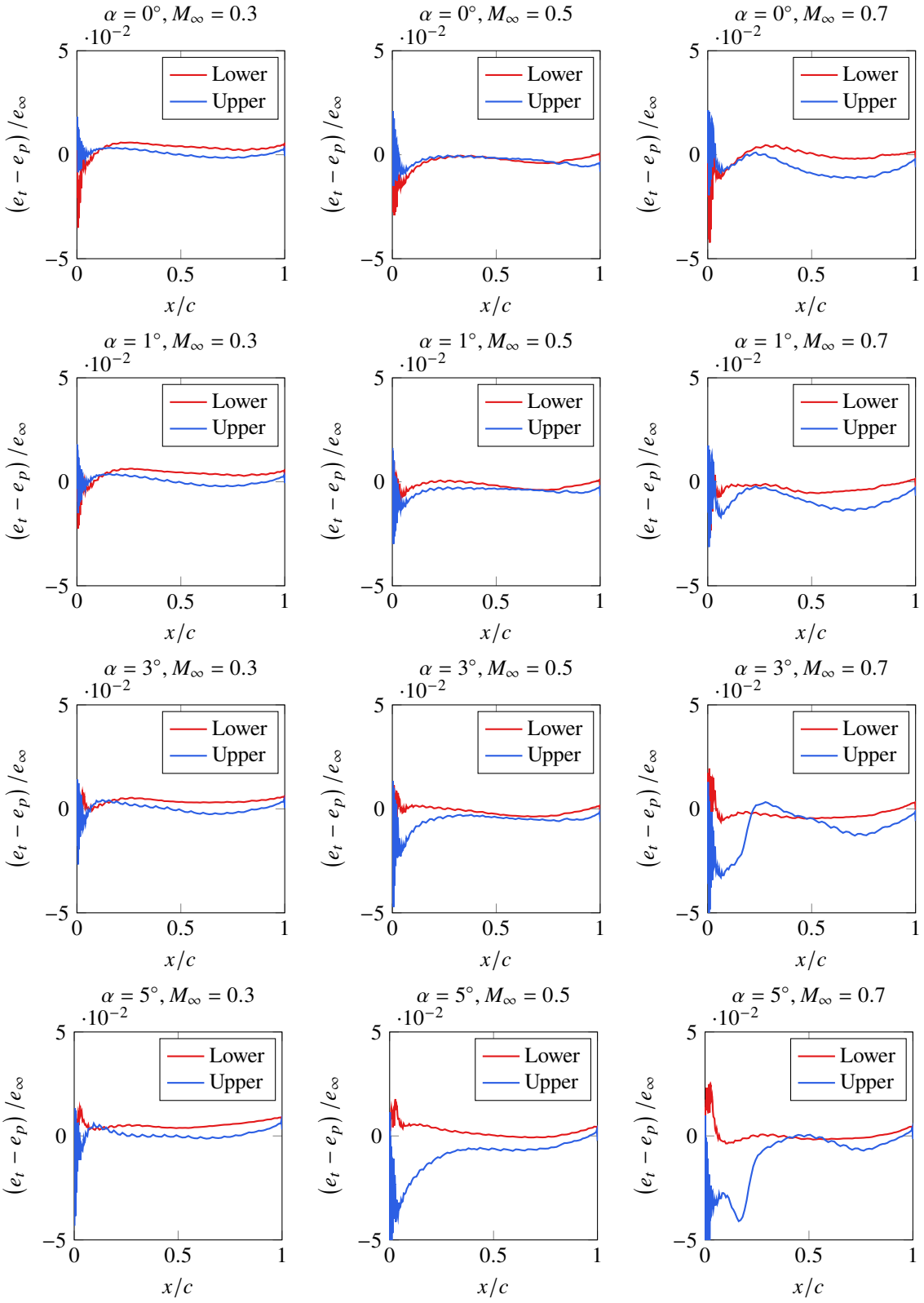


Fig. 7 Comparison of the normalized surface energy error between predictions of the surrogate model and HF CFD. The normalized error is plotted against the normalized coordinate (x/c) from the leading edge to the trailing edge of the airfoil. Results are organized by test runs, with M_∞ of $0.3, 0.5, 0.7$ (from left to right) and α_∞ of $0°, 1°, 3°, 5°$ (from top to bottom).

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

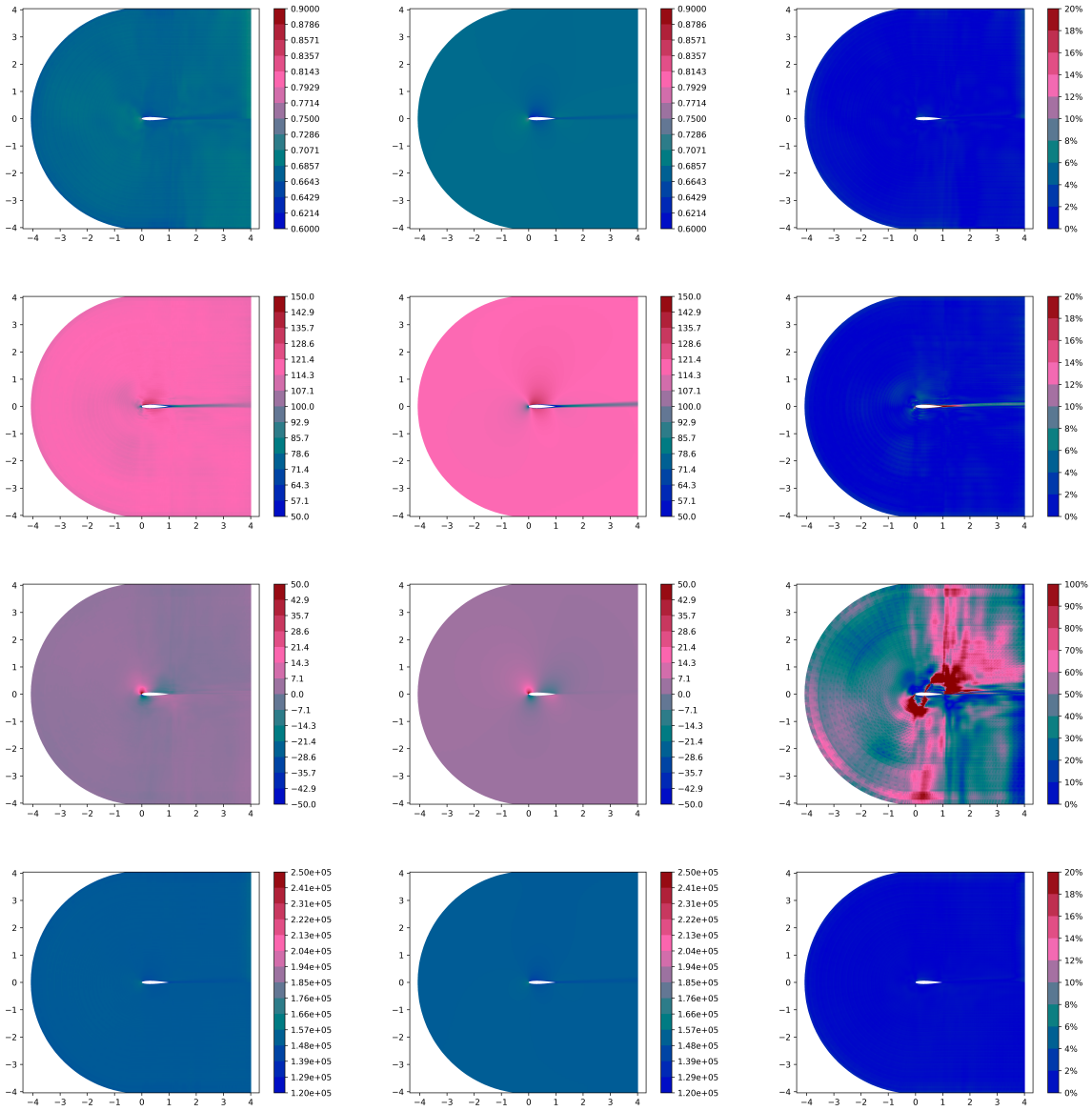


Fig. 8 Displayed from top to bottom are representations of the flow field for density, x-momentum, y-momentum, and energy. In the sequence, the panels show the predicted HF flow field ($q^{\mathbf{HF}}_{\mathbf{pred}}$), the reference value obtained from HF CFD ($q^{\mathbf{HF}}$), and the relative error between $q^{\mathbf{HF}}_{\mathbf{pred}}$ and $q^{\mathbf{HF}}$, arranged from left to right.

C. Aerodynamic Database

To showcase the potential multi-query application of the proposed surrogate model for predicting aerodynamic performance, we characterize the C_L and C_D, computed across a range of Mach numbers and angles of attack. The C_L and C_D are computed for LF CFD, HF CFD, and surrogate model predictions, derived from surface energy according to (6). The calculations are performed at $M_\infty \in \{0.3, 0.5, 0.7\}$ and $\alpha_\infty \in \{0.0, 1.0, 2.0, 3.0, 4.0, 5.0\}^\circ$ to cover the operational range.

For each fidelity source, we employ cubic surface interpolation [38] to generate a continuous surface representation. This method produces cubic splines with minimal surface curvature, providing a concise representation of the database based on scatter data inputs. Figure 9 visually presents the LF Euler CFD database by a blue surface, the HF RANS

CFD database (also the reference in this study) by a green surface, and the surrogate model predictions by a red surface. Black triangles, overlaid on the red surface, represent predicted C_L at corresponding α_∞ and M_∞, emphasizing the accuracy of the interpolated surface.

Analyzing the C_L plot on the left, we observe that the LF CFD predictions are reasonably accurate at lower Mach numbers with an absolute relative error less than 8.6 %. However, at higher Mach numbers, the LF CFD deviates significantly from the reference, particularly at $\alpha_\infty = 5^\circ$ an $M_\infty = 0.7$, with a maximum deviation of 33.9 %, likely due to inaccurate shockwave location predictions. On the other hand, the surrogate model consistently under-predicts C_L independent of α_∞ and M_∞, showing an average deviation of 20.2 % below the reference. The reasons for this deviation, in addition to the previously discussed unsatisfactory convergence of momentum fields and larger surface energy errors at the leading edge, are currently under investigation.

Turning to the C_D predictions on the right, the LF CFD predictions deviate more significantly from the HF reference, a trend expected from Euler simulation results. The relative error peaks at 150.9 % for $\alpha_\infty = 0^\circ$ and $M_\infty = 0.3$, and drops to 10.5 % for $\alpha_\infty = 5^\circ$ and $M_\infty = 0.5$. In contrast, the surrogate model exhibits more robust C_D predictions across the operational range, closely following the reference surface. The average prediction error is an under-prediction of 6.3 %, with a maximum deviation of 25.1 % observed at $\alpha_\infty = 1^\circ$ and $M_\infty = 0.7$, and a minimum deviation of 1.3 % at $\alpha_\infty = 1^\circ$ and $M_\infty = 0.3$. Given the smaller error margins in C_D predictions compared to the LF tool, the surrogate model may prove advantageous in certain applications.

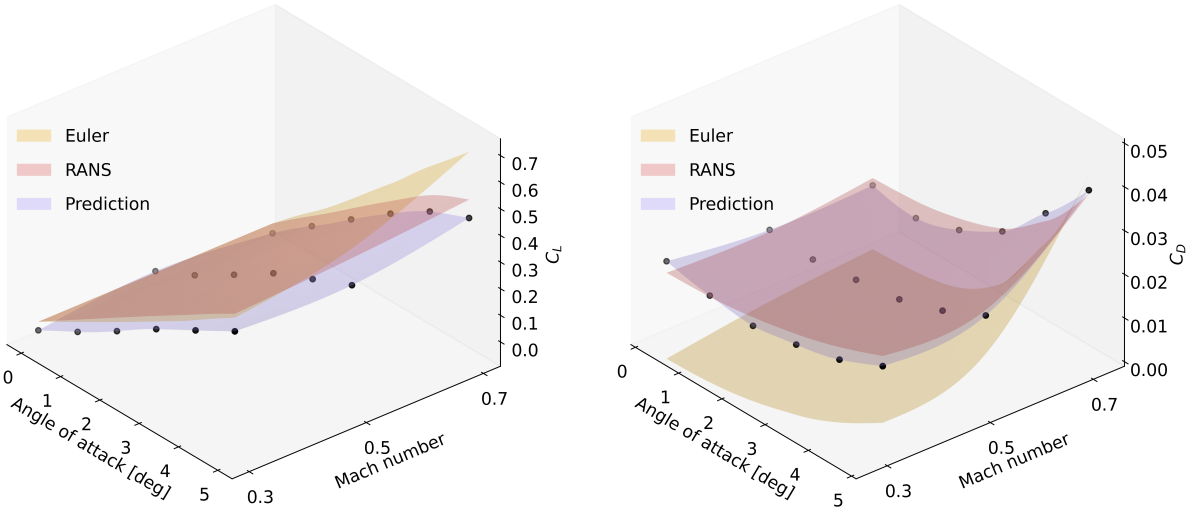


Fig. 9 Aerodynamic database computed from LF Euler CFD (yellow), HF RANS CFD (red) and HF surrogate model (purple). The black circles represent the corresponding coefficient calculated with the surrogate model.

The RMSE error for all test points is quantified and presented in Table 2. To facilitate a comparison of the computational cost required to generate the database shown in Figure 9, we list the total core-hours for running the LF, HF, and surrogate models on the same 2.4 GHz CPU. The results indicate that the proposed model yields an overall error for C_L comparable to that of the LF CFD, while significantly reducing the overall error for C_D. Moreover, the computation cost for the proposed model is three orders of magnitude less than that required for LF CFD, and even more so for HF. This substantial reduction in computation time underscores the advantages of adopting the proposed method in scenarios requiring multi-query capabilities or real-time computational applications. In considering the overall expenditure for constructing the proposed surrogate model, one can break down the total cost into two main components: data generation and model training. The data generation process entails 153 q^{LF}, 153 q^{MF}, and 44 q^{HF} computations. Specifically, the q^{LF} and q^{MF} data generation incurs a computational cost of 4.6 core-hours, while generating the q^{HF} dataset demands 46.5 core-hours. Additionally, the surrogate model's training requires 21 min on a GPU. A significant portion of the cost is attributed to the HF CFD computations, whereas the generation of LF data is notably scalable, particularly when utilizing parallel computing. Although under the test case application of the aerodynamic database presented in Figure 9, the construction cost of the surrogate model exceeds the expense of conducting HF CFD computations (given that only 18 data queries were employed in this instance), it is believed that

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

for applications necessitating a larger number of queries, the overall cost of training such a surrogate model could be more economical than conducting HF CFD computations.

Method	Prediction Cost [core-hrs]	RMSE C_L	RMSE C_D
LF Euler CFD	5.51×10^{-1}	0.0540	0.0153
Proposed Model	7.78×10^{-4}	0.0528	0.0020
HF RANS CFD	1.90×10^{1}	–	–

Table 2 Summary of the prediction computational costs and their corresponding accuracy for different methods.

V. Conclusion and Future Work

This study effectively demonstrates the development of an efficient ML-based surrogate model for aerodynamics. Employing an autoencoder-decoder architecture with super-resolution capability, coupled with multi-fidelity data encompassing both physical modeling and spatial resolution, the methodology utilizes a compact network design. This facilitates training with a reduced dataset size through the incorporation of TL techniques, expediting the overall training cost. The resulting surrogate model produces flow field and aerodynamic coefficient predictions with significantly reduced computational costs compared to traditional CFD methods, while maintaining comparable prediction quality between LF and HF CFD predictions. For multi-query applications requiring extensive HF datasets, the combined cost of building LF, MF, and HF datasets and training the surrogate model can be lower than conducting HF CFD simulations.

However, the study reveals limitations in flow field prediction accuracy and model generalizability. Numerical artifacts arise during the super-resolution step when transitioning from lower to higher spatial resolutions. Additionally, as the model relies on a data-driven approach without physical constraints, the reconstructed flow may violate conservation laws, and the absence of encoded spatial information in the autoencoder limits adaptability to different geometrical configurations.

Future research will systematically address these critical aspects to augment the model's efficacy across a broader spectrum of aerodynamic applications. Addressing the mapping between spatial resolutions is crucial to minimize artificial noise in the predictions. The ongoing exploration of alternative architectures, striking a balance between robustness and compactness, holds promise. Improving flow field prediction accuracy, especially within boundary layers, necessitates the integration of physical constraints into the loss function, incorporating physical conservation laws. Finally, the anticipated inclusion of geometric information encoding in the surrogate model aims to bolster its generalizability and adaptability to diverse geometries in aerodynamic design.

VI. Acknowledgements

The authors extend their gratitude to the Stanford Research Computing Center for the provision of computational resources on the Sherlock cluster. Additionally, our thanks go to the Google Cloud Research Credit program for granting access to GPU resources on the Google Cloud Platform.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

References

[1] Rizzi, F., Blonigan, P. J., Parish, E. J., and Carlberg, K. T., "Pressio: Enabling projection-based model reduction for large-scale nonlinear dynamical systems," , 2020. https://doi.org/10.48550/ARXIV.2003.07798, URL https://arxiv.org/abs/2003.07798.

[2] Brunton, S. L., Noack, B. R., and Koumoutsakos, P., "Machine Learning for Fluid Mechanics," Annual Review of Fluid Mechanics, Vol. 52, 2020, pp. 477–508. https://doi.org/10.1146/annurev-fluid-010719-060214.

[3] Mao, Z., Jagtap, A. D., and Karniadakis, G. E., "Physics-informed Neural Networks for High-speed Flows," Computer Methods in Applied Mechanics and Engineering, Vol. 360, 2020, p. 112789. https://doi.org/10.1016/j.cma.2019.112789, URL https://doi.org/10.1016/j.cma.2019.112789.

[4] Lee, K., and Carlberg, K. T., "Model reduction of dynamical systems on nonlinear manifolds using deep convolutional autoencoders," Journal of Computational Physics, Vol. 404, 2020, p. 108973. https://doi.org/10.1016/J.JCP.2019.108973.

[5] Ng, L. W.-T., and Eldred, M., "Multifidelity Uncertainty Quantification Using Non-Intrusive Polynomial Chaos and Stochastic Collocation," 53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference, 2012. https://doi.org/10.2514/6.2012-1852, URL https://arc.aiaa.org/doi/abs/10.2514/6.2012-1852.

[6] Ng, L. W. T., and Willcox, K. E., "Multifidelity approaches for optimization under uncertainty," International Journal for Numerical Methods in Engineering, Vol. 100, No. 10, 2014, pp. 746–772. https://doi.org/https://doi.org/10.1002/nme.4761, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/nme.4761.

[7] Zhang, K., and Han, Z., "Support Vector Regression-based Multidisciplinary Design Optimization in Aircraft Conceptual Design," 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, 2013. https://doi.org/10.2514/6.2013-1160, URL https://arc.aiaa.org/doi/abs/10.2514/6.2013-1160.

[8] Yondo, R., Andres, E., and Valero, E., "A review on design of experiments and surrogate models in aircraft real-time and many-query aerodynamic analyses," Progress in Aerospace Sciences, Vol. 96, 2018, pp. 23–61. https://doi.org/https://doi.org/10.1016/j.paerosci.2017.11.003, URL https://www.sciencedirect.com/science/article/pii/S0376042117300611.

[9] Carlberg, K. T., Jameson, A., Kochenderfer, M. J., Morton, J., Peng, L., and Witherden, F. D., "Recovering Missing CFD data for High-order Discretizations using Deep Neural Networks and Dynamics Learning," Journal of Computational Physics, Vol. 395, 2019, pp. 105–124. https://doi.org/10.1016/j.jcp.2019.05.041.

[10] Jiang, C. M., Esmaeilzadeh, S., Azizzadenesheli, K., Kashinath, K., Mustafa, M., Tchelepi, H. A., Marcus, P., Prabhat, and Anandkumar, A., "MeshfreeFlowNet: A Physics-Constrained Deep Continuous Space-Time Super-Resolution Framework," Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE Press, 2020.

[11] Obiols-Sales, O., Vishnu, A., Malaya, N., and Chandramowliswharan, A., "CFDNet: A Deep Learning-Based Accelerator for Fluid Simulations," Proceedings of the 34th ACM International Conference on Supercomputing, Association for Computing Machinery, New York, NY, USA, 2020. https://doi.org/10.1145/3392717.3392772, URL https://doi.org/10.1145/3392717.3392772.

[12] Obiols-Sales, O., Vishnu, A., Malaya, N. P., and Chandramowlishwaran, A., "SURFNet: Super-Resolution of Turbulent Flows with Transfer Learning using Small Datasets," 2021 30th International Conference on Parallel Architectures and Compilation Techniques (PACT), 2021, pp. 331–344. https://doi.org/10.1109/PACT52795.2021.00031.

[13] Yu, J., and Hesthaven, J. S., "Flowfield Reconstruction Method Using Artificial Neural Network," AIAA Journal, Vol. 57, No. 2, 2019, pp. 482–498. https://doi.org/10.2514/1.J057108, URL https://doi.org/10.2514/1.J057108.

[14] Meng, X., and Karniadakis, G. E., "A composite neural network that learns from multi-fidelity data: Application to function approximation and inverse PDE problems," Journal of Computational Physics, Vol. 401, 2020, p. 109020. https://doi.org/https://doi.org/10.1016/j.jcp.2019.109020, URL https://www.sciencedirect.com/science/article/pii/S0021999119307260.

[15] Kou, J., Ning, C., and Zhang, W., "Transfer Learning for Flow Reconstruction Based on Multifidelity Data," AIAA Journal, Vol. 60, No. 10, 2022, pp. 5821–5842. https://doi.org/10.2514/1.J061647, URL https://doi.org/10.2514/1.J061647.

[16] Kou, J., and Zhang, W., "Data-driven modeling for unsteady aerodynamics and aeroelasticity," Progress in Aerospace Sciences, Vol. 125, 2021, p. 100725. https://doi.org/https://doi.org/10.1016/j.paerosci.2021.100725, URL https://www.sciencedirect.com/science/article/pii/S0376042121000300.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

[17] Dupuis, R., Jouhaud, J.-C., and Sagaut, P., "Aerodynamic Data Predictions for Transonic Flows via a Machine-Learning-based Surrogate Model," 2018 AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, 2018. https://doi.org/10.2514/6.2018-1905, URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-1905.

[18] Sun, L., Gao, H., Pan, S., and Wang, J.-X., "Surrogate modeling for fluid flows based on physics-constrained deep learning without simulation data," Computer Methods in Applied Mechanics and Engineering, Vol. 361, 2020, p. 112732. https://doi.org/https://doi.org/10.1016/j.cma.2019.112732, URL https://www.sciencedirect.com/science/article/pii/S004578251930622X.

[19] Li, K., Kou, J., and Zhang, W., "Deep Learning for Multifidelity Aerodynamic Distribution Modeling from Experimental and Simulation Data," AIAA Journal, Vol. 60, No. 7, 2022, pp. 4413–4427. https://doi.org/10.2514/1.J061330, URL https://doi.org/10.2514/1.J061330.

[20] Patil, A., Viquerat, J., Larcher, A., El Haber, G., and Hachem, E., "Robust deep learning for emulating turbulent viscosities," Physics of Fluids, Vol. 33, No. 10, 2021, p. 105118. https://doi.org/10.1063/5.0064458, URL https://doi.org/10.1063/5.0064458.

[21] Song, D. H., and Tartakovsky, D. M., "Transfer Learning on Multi-Fidelity Data," Journal of Machine Learning for Modeling and Computing, Vol. 3, No. 1, 2022, pp. 31–47. https://doi.org/10.48550/ARXIV.2105.00856, URL https://arxiv.org/abs/2105.00856.

[22] Pan, S. J., Tsang, I. W., Kwok, J. T., and Yang, Q., "Domain Adaptation via Transfer Component Analysis," IEEE Transactions on Neural Networks, Vol. 22, No. 2, 2011, pp. 199–210. https://doi.org/10.1109/TNN.2010.2091281.

[23] Liao, P., Song, W., Du, P., and Zhao, H., "Multi-fidelity convolutional neural network surrogate model for aerodynamic optimization based on transfer learning," Physics of Fluids, Vol. 33, No. 12, 2021, p. 127121. https://doi.org/10.1063/5.0076538, URL https://doi.org/10.1063/5.0076538.

[24] Spalart, P. R., and Allmaras, S. R., "A One-equation Turbulence Model for Aerodynamic Flows," Recherche Aerospatiale, , No. 1, 1994, pp. 5–21. https://doi.org/10.2514/6.1992-439.

[25] Palacios, F., Alonso, J., Duraisamy, K., Colonno, M., Hicken, J., Aranake, A., Campos, A., Copeland, S., Economon, T., Lonkar, A., et al., "Stanford University Unstructured (SU2): an Open-source Integrated Computational Environment for Multi-physics Simulation and Design," 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition, 2013, p. 287.

[26] Palacios, F., Economon, T. D., Aranake, A., Copeland, S. R., Lonkar, A. K., Lukaczyk, T. W., Manosalvas, D. E., Naik, K. R., Padron, S., Tracey, B., et al., "Stanford University Unstructured (SU2): Analysis and Design Technology for Turbulent Flows," 52nd Aerospace Sciences Meeting, 2014, p. 0243.

[27] Economon, T. D., Palacios, F., Copeland, S. R., Lukaczyk, T. W., and Alonso, J. J., "SU2: An Open-Source Suite for Multiphysics Simulation and Design," AIAA Journal, Vol. 54, No. 3, 2016. https://doi.org/10.2514/1.J053813.

[28] Glorot, X., and Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks," Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research, Vol. 9, edited by Y. W. Teh and M. Titterington, PMLR, Chia Laguna Resort, Sardinia, Italy, 2010, pp. 249–256. URL https://proceedings.mlr.press/v9/glorot10a.html.

[29] Fei, N., Gao, Y., Lu, Z., and Xiang, T., "Z-Score Normalization, Hubness, and Few-Shot Learning," 2021 IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 142–151. https://doi.org/10.1109/ICCV48922.2021.00021.

[30] Wang, X., Kou, J., and Zhang, W., "Multi-fidelity surrogate reduced-order modeling of steady flow estimation," International Journal for Numerical Methods in Fluids, Vol. 92, No. 12, 2020, pp. 1826–1844. https://doi.org/https://doi.org/10.1002/fld.4850, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/fld.4850.

[31] de Leeuw den Bouter, M. L., Ippolito, G., O'Reilly, T. P. A., Remis, R. F., van Gijzen, M. B., and Webb, A. G., "Deep learning-based single image super-resolution for low-field MR brain images," Scientific Reports, Vol. 12, No. 1, 2022, p. 6362. https://doi.org/10.1038/s41598-022-10298-6, URL https://doi.org/10.1038/s41598-022-10298-6.

[32] Mannam, V., and Howard, S., "Small training dataset convolutional neural networks for application-specific super-resolution microscopy," J Biomed Opt, Vol. 28, No. 3, 2023, p. 036501.

[33] Shi, W., Caballero, J., Huszar, F., Totz, J., Aitken, A. P., Bishop, R., Rueckert, D., and Wang, Z., "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Computer Society, Los Alamitos, CA, USA, 2016, pp. 1874–1883. https://doi.org/10.1109/CVPR.2016.207, URL https://doi.ieeecomputersociety.org/10.1109/CVPR.2016.207.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013

[34] Dong, C., Loy, C. C., He, K., and Tang, X., "Learning a Deep Convolutional Network for Image Super-Resolution," Computer Vision – ECCV 2014, edited by D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Springer International Publishing, Cham, 2014, pp. 184–199.

[35] Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., and He, Q., "A Comprehensive Survey on Transfer Learning," Proceedings of the IEEE, Vol. 109, 2019, pp. 43–76. URL https://api.semanticscholar.org/CorpusID:207847753.

[36] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," Advances in Neural Information Processing Systems 32, Curran Associates, Inc., 2019, pp. 8024–8035. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[37] Kingma, D. P., and Ba, J., "Adam: A Method for Stochastic Optimization," 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, edited by Y. Bengio and Y. LeCun, 2015. URL http://arxiv.org/abs/1412.6980.

[38] SciPy Developers, "SciPy griddata Documentation," https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.griddata.html, 2023. Accessed: 2023-12-09.

Downloaded by Stanford University on May 16, 2024 | http://arc.aiaa.org | DOI: 10.2514/6.2024-0013