POLITECNICO DI TORINO
Repository ISTITUZIONALE

Shape Optimization of a Diffusive High-Pressure Turbine Vane Using Machine Learning Tools

(Article begins on next page)

24 November 2024

Shape Optimization of a Diffusive High-Pressure Turbine Vane Using Machine Learning Tools

Rosario Nastasi, Giovanni Labrini, Simone Salvadori * and Daniela Anna Misul

Dipartimento Energia, Politecnico di Torino, Corso Duca degli Abruzzi, 24, 10124 Torino (TO), Italy; rosario.nastasi@polito.it (R.N.); giovanni.labrini@polito.it (G.L.); daniela.misul@polito.it (D.A.M.)
* Correspondence: simone.salvadori@polito.it

Abstract: Machine learning tools represent a key methodology for the shape optimization of complex geometries in the turbomachinery field. One of the current challenges is to redesign High-Pressure Turbine (HPT) stages to couple them with innovative combustion technologies. In fact, recent developments in the gas turbine field have led to the introduction of pioneering solutions such as Rotating Detonation Combustors (RDCs) aimed at improving the overall efficiency of the thermodynamic cycle at low overall pressure ratios. In this study, a HPT vane equipped with diffusive endwalls is optimized to allow for ingesting a high-subsonic flow ($Ma = 0.6$) delivered by a RDC. The main purpose of this paper is to investigate the prediction ability of machine learning tools in case of multiple input parameters and different objective functions. Moreover, the model predictions are used to identify the optimal solutions in terms of vane efficiency and operating conditions. A new solution that combines optimal vane efficiency with target values for both the exit flow angle and the inlet Mach number is also presented. The impact of the newly designed geometrical features on the development of secondary flows is analyzed through numerical simulations. The optimized geometry achieved strong mitigation of the intensity of the secondary flows induced by the main flow separation from the diffusive endwalls. As a consequence, the overall vane aerodynamic efficiency increased with respect to the baseline design.

Keywords: turbomachinery; computational fluid dynamics; machine learning; artificial neural network; random forest; aerodynamics; optimization; genetic algorithm; rotating detonation combustion

1. Introduction

Numerical optimization has been widely applied during the aerodynamic design of High-Pressure Turbine (HPT) components. In this context, Machine Learning (ML) tools play a key role in identifying complex relations between input variables and output objectives [1], providing fast predictions after an adequate training phase. Several studies proved that estimations from Computational Fluid Dynamics (CFD) can be used to collect reliable data for training surrogate models [2–5]. Recent approaches suggest faster solutions that merge the knowledge from an existing dataset of 2D simulations with high-fidelity 3D CFD simulations [6], thus reducing the overall number of simulations required. Among the machine learning tools, Artificial Neural Networks (ANN) are usually preferred for their flexible architecture and wide versatility. In the aerodynamic optimization context, Mengistu and Ghaly [7] used ANN as a surrogate model to optimize a transonic turbine vane and a subsonic compressor rotor in terms of adiabatic efficiency and pressure loss coefficient. Zhang et al. [8] instead predict the airfoil lift coefficient with a convolutional neural network. Du et al. [9] proposed a deep-learning tool for performance prediction and turbine blade profile design. This methodology uses a dual Convolutional Neural Network (CNN) trained over Reynolds-Averaged Navier–Stokes (RANS) simulations to predict the physical field distribution based on design variables and recognize aerodynamic performance parameters from physical field information. Random Forest (RF) models also represent

a valid alternative to an ANN for regression problems, even if they are used more rarely. However, Dasari et al. [10] proved that if RF undergoes proper hyperparameter tuning, it can be used as a surrogate model to support design space exploration. Hyperparameter optimization represents a fundamental step for both defining the ideal architecture of a metamodel and calibrating the metamodel's parameters based on the specific problem that must be solved. Among the different techniques, Bergstra and Bengio [11] demonstrated that random search can find models that are similar or even better than the one found by grid search, within a small fraction of the computational time. Once trained and tuned, surrogate machine learning models can strongly facilitate design optimization by reducing the necessary prediction time of the desired output. Thanks to this property, metamodels are often combined with evolutionary optimization algorithms such as genetic algorithms to identify the maximum or the minimum within the design space. Genetic algorithms are biologically inspired optimization methods that follow the principle of selection, crossover, and mutation to find the best individual starting from a large initial population. Thanks to these characteristics, they are widely recognized as excellent methods for hard optimization problems [12]. This strategy is usually contrasted with gradient-based approaches (i.e., adjoint methods), which use deterministic methods and exploit the information of the gradient of the output parameter with respect to the input variables to produce a progressive increase in the objective function at each iteration of the process. Gradient-based approaches have the advantage of converging within a limited number of iterations but the optimization can easily stop in a local optimum if the initial solution is not chosen carefully [13,14]. Apart from optimization purposes, a properly trained machine learning model is able to provide fast and accurate predictions for any value of the input data within the training design range, thus avoiding the computational effort of a traditional numerical simulation used to predict the desired output.

Component interaction has always been a paramount topic in turbomachinery due to the coupling between a reactive module and the HPT stage [15]. This is especially true in Rotating Detonation Engines (RDE), where the unsteady supersonic combustion products are entrained into the turbine vane, which is usually designed for a steady inflow. Therefore, particular interest should be dedicated to the aerodynamic optimization of HPT vanes for RDE) applications [16], as they are particularly subjected to strong spatial and temporal variations of the inlet conditions. In this context, Liu et al. [17] studied two axial turbine designs exposed to a pulsating inlet with an inlet Mach number of 0.3 and 0.6 through Unsteady Reynolds-Averaged Navier–Stokes (URANS) simulations. They concluded that the aerodynamic efficiency is significantly penalized for an inlet Mach number of 0.6 due to the local flow separations induced by the endwall diffusion. Moreover, for constant endwall shape the turbine results unstarted. As regards the total pressure damping, the $Ma = 0.6$ solution provides better attenuation. A further optimization step was presented by Grasa and Paniagua [18], who parametrized a diffusive vane in the high-subsonic regime to reduce pressure loss and averaged pressure distortion. This analysis considers also three different inlet angles to account for the effect of the vane incidence. Later, Gallis et al. [19] proposed a parametric optimization for both the diffusive endwalls and the airfoil including a flow control system through an array of cooling holes located upstream of the leading edge. The effect of the flow control system was to mitigate the oscillating inflow and to better guide the outflow for the subsequent rotor.

The current work proposes an efficient optimization process for designing an HPT vane that operates at $Ma_1 = 0.6$ using machine learning tools such as ANN and RF. The original blade and diffusive endwall profiles are first parametrized using splines and control points, then each parameter is varied within its design range to achieve new configurations. Subsequently, the machine learning tools are trained on a Design Of Experiments (DOE) composed of 885 samples selected through the Latin Hypercube Sampling (LHS) approach. Each sample is tested with a RANS simulation aimed at calculating the vane efficiency, the exit flow angle, and the inlet Mach number. Based on these outputs, the metamodel hyperparameters are tuned with a Random Search (RS) approach to optimize the prediction

accuracy. The ANN and the RF are then coupled with a genetic algorithm that identifies the optimal solution. The effects of the optimized geometrical features on the internal flow field are also extensively discussed.

2. Parametric Design and Data Collection

The vane profile that is used as a baseline case for the optimization is the one described by Sieverding et al. [20] and analyzed by Denos et al. [21]. However, even though the original vane by Sieverding et al. [20] was inserted into a straight annular channel, in the present work the endwalls are diffusive to allow for a higher inlet Mach number value of ≈0.6 with respect to the original working condition of ≈0.2. The nominal endwall configuration considered in this activity is the one located at the center of the design space described by Gallis et al. [19]. The baseline numerical domain is represented in Figure 1.
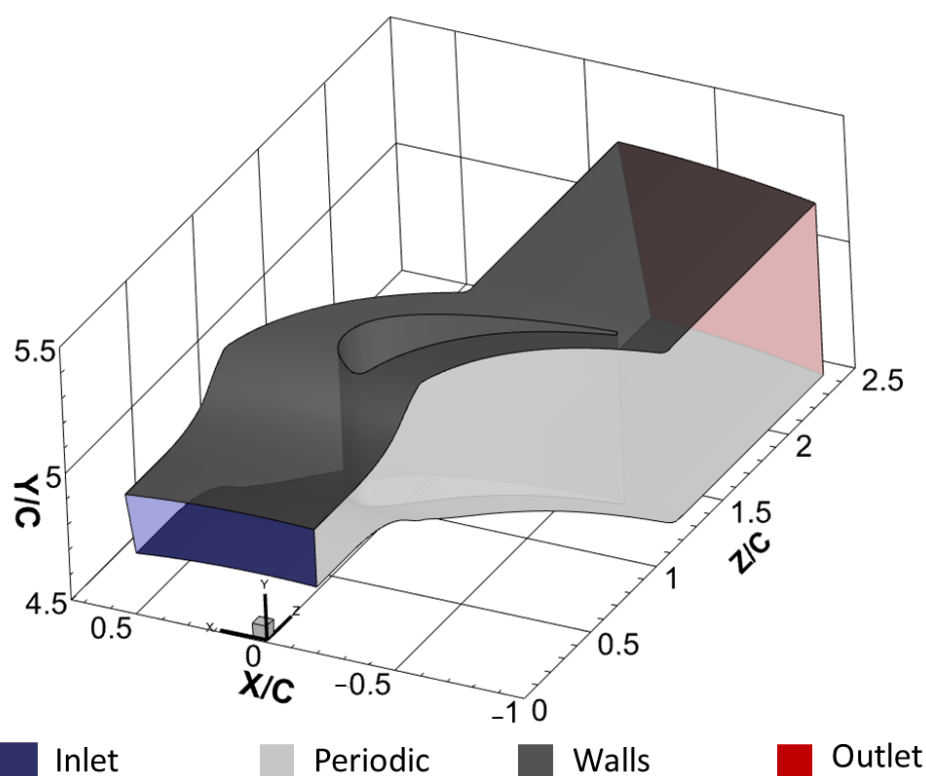

Figure 1. Baseline numerical domain.

The lateral view of the coupled geometry with the endwall and the vane is visible in Figure 2, while the nominal profile is visible from the top of the channel in Figure 2b. Both figures also show the control points of the splines used to create both the vane and the diffusive endwall profiles. More specifically, black points are fixed, red points can only translate in one direction (either horizontally or vertically), and blue points have 2 degrees of freedom and can move both vertically and horizontally. For the sake of simplicity, the endwall profile is considered symmetrical with respect to the mid-span, and the area expansion is controlled by four control points whose positions are free to translate in the Z-direction. The vane profile is deformed by four points on the suction side and three on the pressure side, for an overall number of 14 variables including the stagger angle. The blade profile is later closed using an elliptic curve at the leading edge and a circular arc at the trailing edge, taking care of maintaining a continuous derivative of the curves in the conjunction points. The above-mentioned parametrization was kept equal to the one presented by Gallis et al. [19]. However, the number of samples in the DOE was increased from ≈300 to ≈900 to improve the design space coverage.
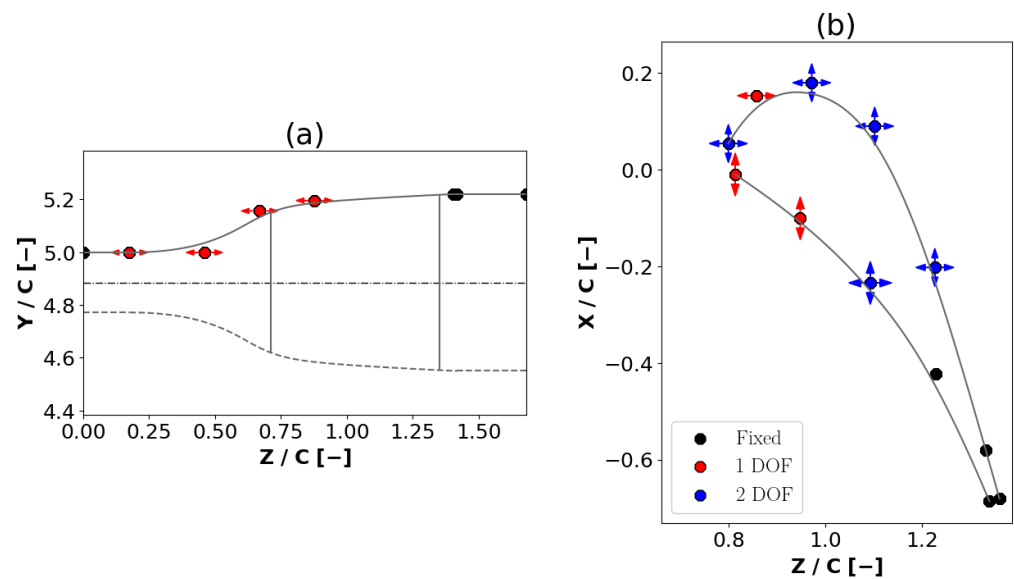
Figure 2. Baseline design: (**a**) endwall profile with control points, and (**b**) vane profile with control points.

The sampling approach used to create the DOE is the Latin Hypercube Sampling (LHS). LHS was originally proposed by McKay et al. [22] and is a nearly random statistical instrument used to generate samples in case of a multi-dimensional design space. It consists of dividing the range of variability of each design variable X_k into N sub-intervals of equal marginal probability $\frac{1}{N}$, and randomly sampling once from each sub-interval. In the current work, the number of samples and the corresponding sub-intervals were selected to be equal to $N = 900$ to obtain an extensive coverage of the entire design space. Out of the 900 samples, 885 design points successfully completed both the automatic meshing process and the calculation, thus producing the overall dataset. The design space exploration for the endwall and the vane profile is represented in Figure 3a,b. The aerodynamic performance of each sample is compared with the one obtained for the baseline geometry that is composed by the nominal airfoil [20] and by the symmetrical diffusive endwall profile located at the center of the design space explored by Gallis et al. [19].
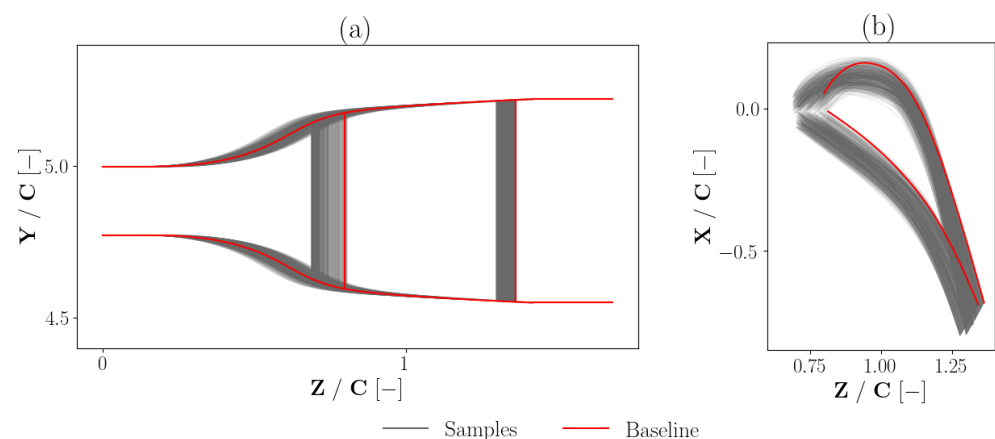

Figure 3. DOE: (**a**) Endwall profiles generated using LHS, and (**b**) Vane profiles generated using LHS.

3. Numerical Methodology

In the current work, the commercial solver ANSYS CFX™ (2022R1) is used to run the RANS numerical analysis of the vane. The boundary conditions applied to the domain are summarized in Table 1.

Table 1. Boundary conditions.

Inlet total pressure	161,600 Pa
Inlet total temperature	440 K
Wall conditions	Adiabatic (no-slip)
Periodic surfaces	Angular periodicity
Outlet static pressure	83,289 Pa

It is important to stress that all the simulations are performed at an equal inlet total pressure. As a consequence, the overall mass flow rate changes according to the variation of the minimum throat area. This approach was chosen in order to control the inlet Mach number with the mass flow rate since the objective of the study is to target $Ma_1 = 0.6$. CFD simulations are performed under steady-state conditions. The "high resolution" scheme is adopted for advection and turbulence discretization, while the k-ω SST model is used for turbulence closure. The solver uses a coupled pressure-based approach, and the viscous work term is included in the calculation. The simulation of the baseline domain is performed on an unstructured mesh with $\approx$2,000,000 tetrahedral elements (Figure 4a,b). Furthermore, a detailed view of the 20 inflation layers that are used to keep $y^+ < 1$ in the wall regions is provided in Figure 4c. The mesh size was selected after the evaluation of the Grid Convergence Index (GCI) [23]. This analysis was conducted by considering the inlet mass flow rate, the inlet mass-weighted averaged Mach number, and the mass-weighted averaged total pressure at the outlet of the vane for three different levels of grid size. The refinement ratio between coarse (C), medium (M), and fine (F) mesh was kept equal to $\approx$1.25. The results in Table 2 suggest that the medium refinement provides mesh-independent results with an asymptotic range of convergence ≈ 1. Consequently, the medium level of mesh was adopted for all the CFD simulations presented in this research work. More details about the mesh sensitivity are described by Gallis et al. [19].

Table 2. Grid Convergence Index.

	GCI (C-M)	GCI (M-F)	Asymptotic Range of Convergence
Ma_1	1.4107%	0.92348%	1.029%
$\dot{m}_1$	1.9418%	1.2732%	1.032%
P_2^0	0.064%	0.022%	1.064%
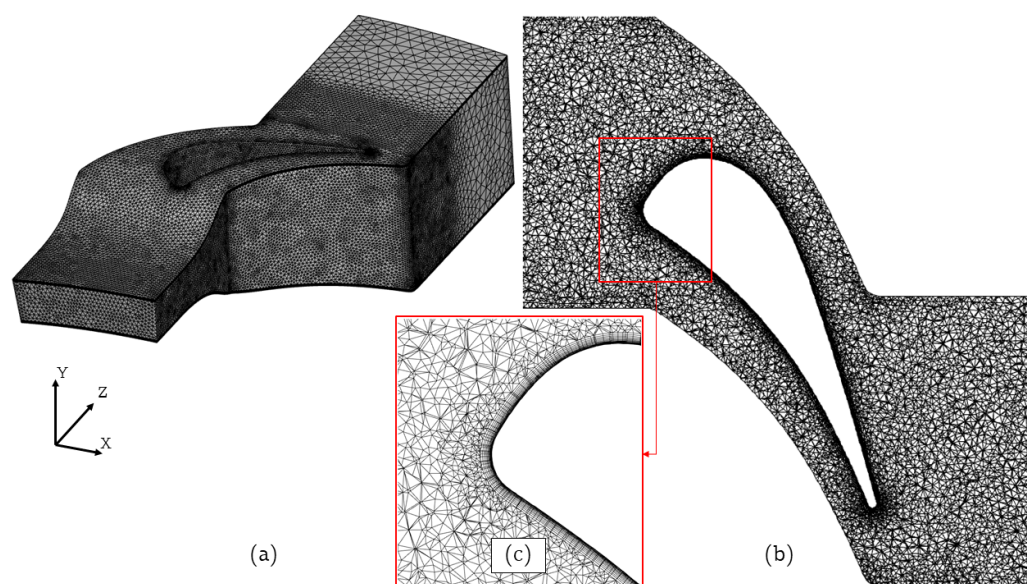
Figure 4. Mesh: (**a**) entire domain, (**b**) mid-span plane, and (**c**) Inflation layers.

The convergence of the simulation is controlled through the maximum number of iterations, which is set equal to 250. This approach ensures residual minimization in the order of 10^{-6}. Additionally, solution reports are used to check the stabilization of the desired outputs during the simulation.

Concerning the numerical method, RANS-based simulations introduce the idea of Reynolds-averaging to split an instantaneous quantity into the sum of a time-averaged and a fluctuating term. The instantaneous velocity U_i is thus decomposed into a time-averaged component $\bar{U}_i$ and a time-varying component u_i. According to this idea, the governing Navier–Stokes equations can be reformulated as in Equation (1) for continuity, Equation (2) for momentum, and Equation (3) for energy.

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho \bar{U}_j\right) = 0 \tag{1}$$

$$\frac{\partial(\rho \bar{U}_i)}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho \bar{U}_i \bar{U}_j\right) = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\tau_{ij} - \rho \overline{u_i u_j}\right) + S_M \tag{2}$$

$$\frac{\partial(\rho h_{tot})}{\partial t} - \frac{\partial p}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho \bar{U}_j h_{tot}\right) = \frac{\partial}{\partial x_j}\left(\Lambda \frac{\partial T}{\partial x_j} - \rho \overline{u_j h}\right) + \frac{\partial}{\partial x_j}\left[\bar{U}_i\left(\tau_{ij} - \rho \overline{u_i u_j}\right)\right] + S_E \tag{3}$$

The τ_{ij} term in Equations (2) and (3) indicates the stress tensor and it is calculated according to Equation (4), while S_M indicates the momentum sources and ρ is the density. The energy Equation (3) is instead formulated considering the total enthalpy h_{tot}, the energy sources are accounted for by the term S_E, and Λ indicates the thermal conductivity.

$$\tau_{ij} = \mu\left(\frac{\partial \bar{U}_i}{\partial x_j} + \frac{\partial \bar{U}_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial \bar{U}_k}{\partial x_k}\right) \tag{4}$$

The Reynolds stresses $\rho \overline{u_i u_j}$ can be instead modeled thanks to the introduction of an appropriate turbulence model, that closes the Reynolds-averaged equations. More specifically, the $k - \omega$ SST model by Menter [24] introduces two transport equations for the turbulent kinetic energy k (Equation (5)) and for the specific turbulent dissipation rate ω (Equation (6)).

$$\frac{\partial(\rho k)}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho \bar{U}_j k\right) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] + P_k - \beta' \rho k \omega + P_{kb} \tag{5}$$

$$\frac{\partial(\rho \omega)}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho \bar{U}_j \omega\right) = \frac{\partial}{\partial x_j}\left[\left(\mu + \frac{\mu_t}{\sigma_k}\right)\frac{\partial \omega}{\partial x_j}\right] + \alpha\frac{\omega}{k}P_k - \beta \rho \omega^2 + P_{\omega b} \tag{6}$$

The term P_k in Equations (5) and (6) is the production rate of turbulence, while if buoyancy model is enabled, P_{kb} and $P_{\omega b}$ are the turbulence buoyancy terms. Additionally, β', α, β, σ_k, and σ_ω are constants whose values can be found in the ANSYS CFX™ theory guide [25]. The turbulent viscosity μ_t is computed from Equation (7) and contains the blending function F_2 (Equation (8)), where y is the distance to the nearest wall, ν is the kinematic viscosity, S is an invariant measure of the strain rate, and $a_1 = 5/9$.

$$\mu_t = \frac{\rho a_1 k}{max(a_1 \omega, SF_2)} \tag{7}$$

$$F_2 = \tanh\left(max\left(\frac{2\sqrt{k}}{\beta' \omega y}, \frac{500\nu}{y^2 \omega}\right)^2\right) \tag{8}$$

Once modeled, μ_t can be finally combined with the Reynolds stresses term using Equation (9).

$$-\rho\overline{u_i u_j} = \mu_t\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) - \frac{2}{3}\delta_{ij}\left(\rho k + \mu_t\frac{\partial U_k}{\partial x_k}\right) \tag{9}$$

Regarding the working fluid, air with an ideal gas hypothesis and constant specific heat c_P is assumed. The aerodynamic performance of the vane is estimated with the vane adiabatic efficiency η in Equation (10), where P_2 and P_2^0 are the static and the total outlet pressure, while P_1^0 is the inlet total pressure and γ is the specific heat ratio.

$$\eta = \frac{1 - \left(\frac{P_2}{P_2^0}\right)^{\frac{\gamma-1}{\gamma}}}{1 - \left(\frac{P_2}{P_1^0}\right)^{\frac{\gamma-1}{\gamma}}} = \frac{u_2^2}{u_{2,is}^2} \tag{10}$$

The efficiency equation can also be expressed by referring to the outlet real velocity u_2 and the outlet isentropic velocity $u_{2,is}$. The efficiency alone gives only an index of the aerodynamic performance of the vane, without caring about the operating conditions of the machine. To overcome this issue, the Root Squared Index Θ was introduced:

$$\Theta = 1 - \sqrt{c_1\left(\frac{Ma_1 - Ma_{1,ref}}{Ma_{1,ref}}\right)^2 + c_2\left(\frac{\eta - \eta_{ref}}{\eta_{ref}}\right)^2 + c_3\left(\frac{\alpha_2 - \alpha_{2,ref}}{\alpha_{2,ref}}\right)^2} \tag{11}$$

Equation (11) combines the deviation of the actual measurement with respect to the reference values in terms of inlet Mach number Ma_1, efficiency, and exit flow yaw angle α_2. More specifically, $Ma_{1,ref} = 0.6$, $\eta_{ref} = 1$ and $\alpha_{2,ref} = 73°$ is the nominal exit metal angle of the vane. Each squared term in Equation (11) was then multiplied by a weight coefficient $c_{1,2,3}$ to ensure equal importance of each objective. Equation (10) for η and Equation (11) for Θ are considered objective functions for two independent optimization processes. In addition, the vane outlet conditions are estimated through the total pressure loss coefficient C_p, which is calculated with Equation (12), where the outlet total and static pressure P_2^0 and P_2 are both measured at an axial distance equal to the 30% of chord C downstream from the vane trailing edge.

$$C_p = \frac{P_1^0 - P_2^0}{P_2^0 - P_2} \tag{12}$$

Moreover, the Contraction Ratio (CR) is introduced during the post-processing step to quantify the variation of the minimum cross-sectional area and it is defined in Equation (13), where A_1 is the inlet area, which is fixed, and A_{th} is the vane throat area that is calculated from Equation (14).

$$CR = \frac{A_1}{A_{th}} \tag{13}$$

$$A_{th} = \frac{\dot{m}\sqrt{RT_1^0}}{P_1^0\sqrt{\gamma\left(\frac{2}{\gamma+1}\right)^{\frac{\gamma+1}{\gamma-1}}}} \tag{14}$$

4. Machine Learning Approaches

4.1. Artificial Neural Network

ANN is the best-known among the different machine learning tools. The network architecture comprises an input layer, one or more hidden layers, and an output layer. Each layer is composed of neurons, that are activated through a specific activation function. A neuron is a mathematical model where weights and biases are calibrated during the training step to obtain the non-linear relationship between input variables and output objectives. Weights are real numbers expressing the importance of the inputs to the output, while biases represent a shift in the activation of a neuron that allows the model to capture

features that are not directly connected to input variables. Each neuron thus processes the input variables as expressed in Equation (15), where z is the weighted sum of the inputs calculated over the total number of input variables N_{var}, w if the weight and b the bias of each input variable x.

$$z = \sum_{i=1}^{N_{var}} w_i x_i + b \qquad (15)$$

The activation function $f(z)$ is then used to decide whether that particular artificial neuron will be activated [26] and also transforms the summed weighted into an output that is transferred to the next hidden layer.

During the training phase, the optimizer minimizes the error between the predicted and the actual outputs by tuning weights and biases. The model is tested on a validation dataset that estimates the network's capability to generalize with data that are not used to train the model. Thanks to the validation test set it is possible to investigate the occurrence of the overfitting or underfitting phenomena. In the case of overfitting, the model only memorizes the training data with limited generalizability. On the contrary, the underfitting demonstrates that the model is too simple and cannot appropriately learn data relationships [27]. The network hyperparameters must, therefore, be appropriately calibrated to avoid these undesired conditions. The parameters selected in this work for the model optimization are the following:

- The number of neurons in each hidden layer.
- The activation function:
    - Hyperbolic tangent function (tanh):

$$f(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \qquad (16)$$

    - Rectified Linear Units (ReLu):

$$f(z) = max(0, z) \qquad (17)$$

    - Leaky Rectified Linear Units (Leaky ReLu):

$$f(z) = max(az, z) \qquad (18)$$

    - Sigmoid:

$$f(z) = \frac{1}{1 + e^{-z}} \qquad (19)$$

Leaky Rectified Linear Units (Leaky ReLu) in Equation (18) is an improvement of the ReLu (Equation (17)) since the latter can suffer from the so-called "dead neurons" problem [28], which means that neurons with negative biases may never be activated. In contrast, both the "tanh" and the Sigmoid functions, respectively, expressed with Equation (16) and in Equation (19), suffer from the "saturation" problem in deep neural network applications [29], when the gradient used to update weights back-propagates from the output layer to the earlier ones.

- The loss function optimizer:
    - Adam : it is an algorithm for first-order gradient-based optimization characterized by an adaptive moment estimation, introduced by Kingma and Ba [30]. This method computes individual adaptive learning rates for different parameters from estimates of the first and second moments of the gradients. During the optimization process, weights are updated inversely proportional to the scaled L^2 norm of past gradients.
    - Adamax: it is an extension of the Adam model where the update rule changes and the L^∞ norm of past gradients is used instead of the L^2 norm.

- Nadam: it combines the Adam optimization algorithm with the idea of Nesterov Accelerated Gradient (NAG) [31].
  - rmsprop: it uses an adaptive learning rate calculated through a moving average of the squared gradient for each weight. In this way, the algorithm provides a faster solution than the stochastic gradient descent approach.

- The dropout rate: it expresses the percentage of neurons that are randomly removed from each layer of the network. In this way, the model becomes simpler and less prone to overfitting [32].
- The L_2 regularization coefficient: regularization is a strategy to prevent overfitting through manipulation of the loss function. L_2, also known as "Ridge regularization", adds a penalty factor to the loss function that is proportional to the squared magnitude of the weight coefficient (Equation (20)). In this way, large coefficients (i.e., large weights) are more penalized, and the influence of a single strong coefficient will be spread across multiple weaker coefficients.

$$\tilde{L}(w) = L(w) + \lambda ||w||^2 \tag{20}$$

In Equation (20), $\tilde{L}$ indicates the regularized loss function, L the original loss function, w the weights and λ the regularization coefficient that controls the strength of the perturbation.

- The batch size: it indicates the number of sub-datasets that are used to train the model.

4.2. Random Forest

RF is a metamodel that fits several decision trees on the training datasets and aggregates their prediction through an averaging procedure [33]. This approach was proposed by Breiman [34] for classification and regression purposes and showed great potential in the case of large numbers of design variables. Each tree in the forest is characterized by a certain number of decision nodes, that split the data according to a proper decision criterion using if/else conditions. The number of decision nodes is proportional to the depth of the tree. Finally, as the maximum depth is reached, the last decision nodes split into the leaf nodes, in which the decision about the class of the instances is made. In regression problems, each decision corresponds to the comparison with a threshold. This condition divides the data into two sub-groups, one greater and one less than the threshold, so the higher the number of decision nodes, the greater the number of sub-ranges into which the initial dataset will be divided. Although this process increases the model's accuracy during the training phase, it can be counterproductive to the ability to generalize during the testing phase, as the model will be more prone to the overfitting phenomenon. In RF applications, "Bootstrap" [35] is used as a resampling approach to reduce the generalization error of the model. The predictive accuracy can be appropriately calibrated by tuning the model hyperparameters:

- Number of estimators: the number of decision trees that create the forest.
- Maximum depth of the tree.
- Minimum sample split: the minimum number of samples required to split a decision node.
- Minimum sample leaf: the minimum number of samples required to be at a leaf node.

The hyperparameter optimization is discussed in Section 4.4.

4.3. Performance Metrics and Loss Function

In the case of regression problems, the model accuracy is monitored by comparing predictions with true values. This comparison can be easily accessed through the Mean Squared Error (*MSE*) value defined in Equation (21) and the coefficient of determination R^2, calculated with Equation (22).

$$MSE = \frac{1}{N} \sum_{j=1}^{N} \left(y_j - \hat{y}_j \right)^2 \tag{21}$$

$$R^2 = 1 - \frac{\sum_{j=1}^{N} \left(\hat{y}_j - \overline{y}_j \right)^2}{\sum_{j=1}^{N} \left(y_j - \overline{y}_j \right)^2} \tag{22}$$

In both equations, N represents the total number of samples, $\hat{y}$ is the model prediction, y is the true value and $\overline{y}$ is its mean value. During the training phase of the model, the MSE is used as a loss function and it is minimized by the model optimizer.

4.4. Hyperparameters Optimization

The optimization of the hyperparameters represents a crucial step in obtaining a reliable predictive model. As far as the ANN is concerned, the model architecture is characterized by an input layer which manages 18 input variables, three hidden layers, and an output layer for the single-objective function. The number of neurons for each hidden layer is tuned using a random search approach. Overall, the hyperparameters optimized through random search and their range of variability are summarized in the first and second columns of Table 3, respectively.

Table 3. ANN hyperparameters.

Hyperparameter	Range	Optimal Model (η)	Optimal Model (Θ)
Number of neurons: first hidden layer	4–128	76	61
Number of neurons: second hidden layer	4–128	81	88
Number of neurons: third hidden layer	4–128	69	47
λ (L_2 regularization): first hidden layer	0–1×10^{-4}	8.4×10^{-5}	2.9×10^{-5}
λ (L_2 regularization): second hidden layer	0–1×10^{-4}	4.4×10^{-5}	9.3×10^{-5}
λ (L_2 regularization): third hidden layer	0–1×10^{-4}	1.3×10^{-5}	5.1×10^{-5}
Dropout rate: first hidden layer	0–30%	3%	0.6%
Dropout rate: second hidden layer	0–30%	12%	30%
Dropout rate: third hidden layer	0–30%	20%	17%
Batch size	8, 16, 32, 64	64	32
Activation function	tanh, ReLu, Sigmoid, Leaky ReLu	Leaky ReLu	Leaky ReLu
Optimizer	Adam, rmsprop, Adamax, Nadam	Adamax	Adamax

Random search is usually preferred over grid search in case of high-dimensional problems, especially when many grid points are required to explore the search space defined by the number of hyperparameters and their range of variability. Moreover, with grid search all points are evenly distributed in the search space, which can produce weak coverage in important sub-ranges and unnecessary coverage in areas of little interest. A deep comparison between random search and grid search is provided by Bergstra and Bengio [11]. In the current work, each combination of hyperparameters was tested using a k-fold cross-validation with $k = 5$. The cross-validation step guarantees that the accuracy of the metamodel is not affected by the train-test split. Finally, the random search approach looks for the combination of hyperparameters that maximize the R^2 coefficient on the test set as an average of the $k = 5$ results. A total number of 500 combinations were tested, resulting in 2500 training phases considering cross-validation too. The number of training

epochs was kept equal to 2000 after making sure that it was a sufficient number for the loss function to stabilize.

The random search approach provides the best values for the model hyperparameters, which are summarized in the last column of Table 3. In both applications, Adamax is preferred as the optimizer and Leaky ReLu as the activation function. The cross-validated results of the best model, which are instead summarized in Table 4, demonstrate that the model accuracy is not affected by the training-test split, as $R^2 \approx 0.99$ independently from the k-fold index.

Table 4. Cross-validation of the optimal ANN model.

k-Fold	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
R^2 (η)	0.988	0.989	0.991	0.988	0.982
R^2 (Θ)	0.997	0.996	0.996	0.997	0.994

A similar approach was then applied to the RF model. The hyperparameters tuned with random search are reported in the first column of Table 5, while the second column indicates their range of variability. As can be seen from the cross-validated results in Table 6, the RF model is not able to reach the same predicting performance as the ANN, and the R^2 coefficient significantly varies from 0.874 to 0.908 at different k-fold index. Due to this lack of accuracy, the RF model was only used to predict η and thus did not undergo further hyperparameters optimization for predicting Θ.

Table 5. RF hyperparameters.

Hyperparameter	Range	Optimal Model (η)
Number of decision trees	5–500	401
Maximum depth	1–20	16
Min samples split	2–10	3
Min samples leaf	1–6	2

Table 6. Cross-validation of the optimal RF model.

k-Fold	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
R^2	0.906	0.874	0.889	0.908	0.884

5. Optimization

The first step of the optimization process consists of training each machine learning model using the best hyperparameters identified during the random search process. This step was performed using a 70%–15%–15% split for train, validation, and test. Starting from the neural network models, the training history for the single and the combined objective function are reported in Figure 5a and Figure 5b, respectively. These graphs show the evolution of training and validation loss with training epochs. In both cases, the models do not suffer from the overfitting phenomenon as the validation loss does not deviate from the training one. Moreover, the *MSE* convergence is approached after approximately 1000 epochs.

More details about the generalization capability of the models can be appreciated by looking at the parity plots in Figure 6a–f. The parity plot of each sub-dataset compares the predictions with the true values, and predictions that perfectly match the true values lie along the diagonal red line. Both the single and the combined objective function problems in Figure 6a–c and in Figure 6d–f, respectively, show a narrow distribution of the samples along the diagonal line. This results in a $R^2 \approx 0.98$ in all sub-dataset of the model used to predict η and $R^2 \approx 0.99$ for the model used to predict Θ. Furthermore, a very important

aspect is the ability to accurately predict the results at the extremes of the parity plot, as these data points represent the area of greatest interest for identifying the minima or maxima of the Objective Function (*OF*).
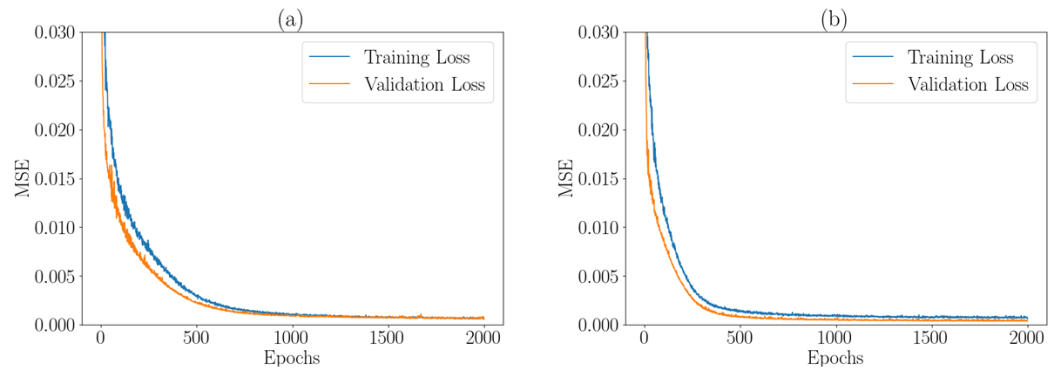

Figure 5. Optimal ANN models training history: (**a**) model used to optimize η, and (**b**) model used to optimize Θ.
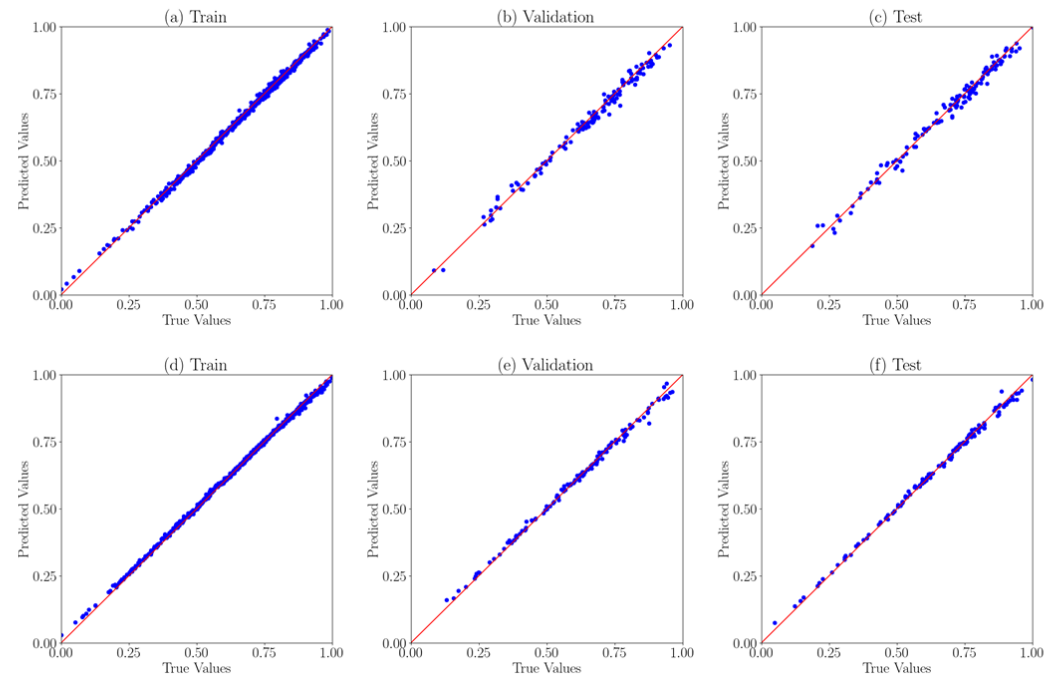

Figure 6. ANN Parity plots: (**a**) training ($R^2 = 0.98$, $OF = \eta$), (**b**) validation ($R^2 = 0.98$, $OF = \eta$), (**c**) test ($R^2 = 0.98$, $OF = \eta$), (**d**) training ($R^2 = 0.99$, $OF = \Theta$), (**e**) validation ($R^2 = 0.99$, $OF = \Theta$), and (**f**) test ($R^2 = 0.99$, $OF = \Theta$).

Similar considerations can also be derived for the RF case. As discussed before, despite the hyperparameter tuning, the optimal RF model does not achieve the same performance as the neural network. The validation and test parity plots above all (Figure 7b,c) show a more sparse distribution of points along the diagonal line, and the R^2 index drops from 0.98 to 0.89 when the model needs to generalize on data that was not used for training. This behavior confirms the cross-validated results reported in Table 6 and adds useful information regarding the presence of overfitting in the RF model.

The lack of accuracy of the RF model can be likely attributable to insufficient data. To prove this idea, the model was trained for an increasing number of samples, and the evolution of the training and test R^2 index shows a progressive improvement of the model generalization ability with increasing data size (Figure 8). However, improvements for

the numbers of samples greater than 600 are quite moderate and the model is not able to achieve the same performance as the ANN, given the same number of samples.
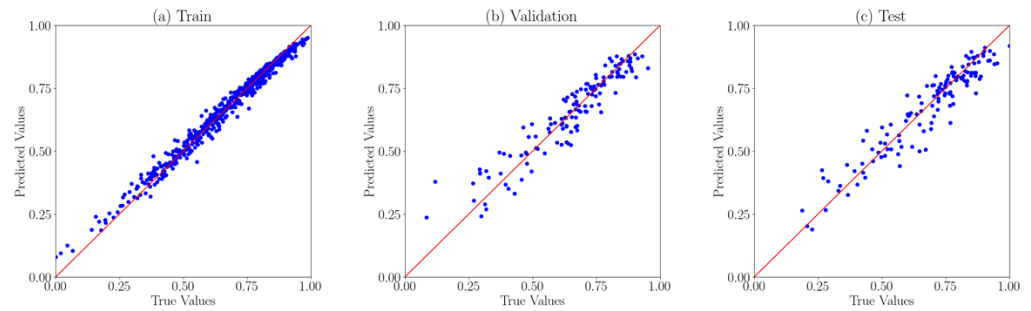

Figure 7. RF Parity plots: (**a**) train ($R^2 = 0.98$), (**b**) validation ($R^2 = 0.88$), and (**c**) test ($R^2 = 0.89$).
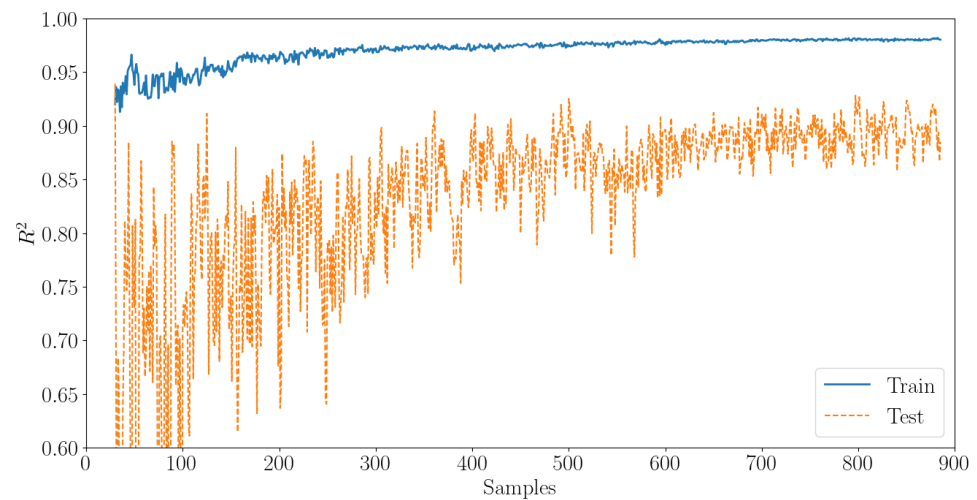

Figure 8. Optimal random forest model trained for different data size.

Once the optimal machine learning model was obtained, it was used to predict the vane geometry that maximizes the single and the combined objective functions. However, the optimization of Θ was conducted only for the ANN case, as the RF model proved to perform worse for this kind of problem. The optimization process uses a genetic algorithm implemented with the Python library PyGAD [36]. The genetic algorithm starts with an initial population of 200 individuals that evolves through 350 consecutive generations. Each individual is represented by a chromosome whose length (number of genes) is proportional to the number of input variables. Performance is instead calculated via a fitness function and the most qualified individuals are selected for mating at each generation. The mating process consists of crossover and mutation. "Crossover" means that the chromosomes are split and swapped between the two parents, while during "mutation" some genes in the chromosome are replaced with random values to increase the disparity between individuals. Eventually, an elitism mechanism is employed to preserve the best 10 individuals of each generation. In the current application, the size of the mating pool was set equal to 120 parents, while 10% of the genes undergo mutation by replacement. The crossover strategy is instead based on a "single-point" approach. Each individual in the genetic algorithm evolution corresponds to a sample whose performance is predicted by the machine learning model. Moreover, the same GA parameters are considered for both the ANN and the RF to obtain their best predictions.

Figure 9a,b reproduces the objective function evolution with the generations, and the optima are identified with a corresponding case ID for simplicity. *OPT-1* and *OPT-2* represent the optimal geometry predicted, respectively, by ANN and RF for η, while *OPT-3* refers to the geometry optimized with ANN in terms of Θ. The plot underlines that most

improvements are achieved within the first 100 generations. Subsequently, only slight enhancements occur until the end of the evolutionary process. The evolution of the objective function in Figure 9a shows a limit of the RF model in exceeding $\eta = 0.9$. This behavior can be confirmed by the lack of accuracy of the model shown in the parity plot (Figure 7), as predictions in the upper extreme of the plot underestimate the corresponding true values.
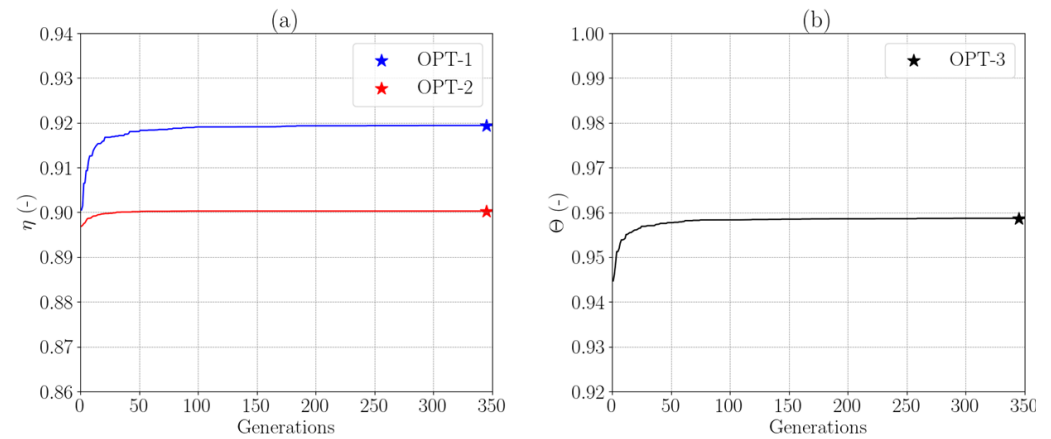

Figure 9. Evolution of the objective function with GA generations: (**a**) optimization of η, and (**b**) optimization of Θ.

The optimized diffusive endwall and vane profiles found by the GA are shown in Figure 10a,b. This graphical comparison reveals that ANN and RF predict the same optimal geometry for maximizing η. Both *OPT-1* and *OPT-2* deviate significantly from the baseline geometry, especially at the leading edge of the vane and at the endwall. The trailing edge instead is almost identical to the original one. Minor differences emerge instead from the comparison of *OPT-1* and *OPT-2* with *OPT-3*, visible only in the first portion of the suction side up to $Z/D = 1$ and in the central zone of the pressure side.
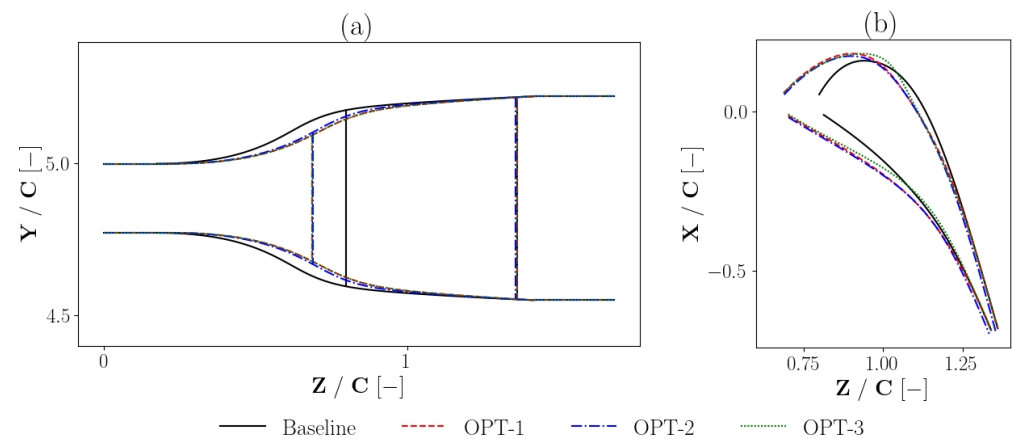

Figure 10. Baseline and optimized geometries: (**a**) diffusive endwalls, and (**b**) vane profile.

The optimal geometries were then tested using RANS simulations to determine the reliability of the predicted objectives and to investigate the impact of these geometrical features on the physics of the problem. Results in terms of model predictions and CFD validation are summarized in Table 7.

Regarding the vane efficiency, the prediction error of both *OPT-1* and *OPT-2* is close to 1%, confirming the two models' great prediction accuracy. CFD validation demonstrated that *OPT-1* is the best solution for η optimization. Given the similarities between *OPT-1* and *OPT-2*, only the first one was examined in depth through post-processing to study the internal flow field. Both the solutions *OPT-1* and *OPT-3* reach the maximum result in terms

of Θ. Moreover, using a combined objective function does not penalize the overall stator efficiency, which is almost unchanged between *OPT-1* and *OPT-3*.

Table 7. GA results and validation with CFD.

Case ID	Model	Prediction	CFD	Error
Baseline	-	-	$\eta = 0.879$	-
		-	$\Theta = 0.932$	-
OPT-1	ANN	$\eta = 0.919$	$\eta = 0.913$	<1%
		-	$\Theta = 0.954$	-
OPT-2	RF	$\eta = 0.900$	$\eta = 0.910$	-1.1%
		-	$\Theta = 0.953$	-
OPT-3	ANN	-	$\eta = 0.912$	-
		$\Theta = 0.958$	$\Theta = 0.954$	<1%

6. CFD Results

In this section, the CFD simulations are post-processed to estimate the impact of the optimized geometries on the internal flow field. The helicity maps in Figure 11a–f shows the intensity of the secondary flows induced by the fluid separation from the walls. A first Separation Bubble (SB) occurs as the endwall starts to diffuse. Subsequently, the impingement with the leading edge produces the horseshoe vortex, which is split into Pressure Side Horseshoe Leg (PSHL) and Suction Side Horseshoe Leg (SSHL). Due to the pressure difference between the pressure and the suction side, the PSHL is rapidly pushed toward the suction side of the adjacent airfoil.

In the baseline geometry (Figure 11a,d), the faster endwall diffusion intensifies the magnitude of the PSHL. The latter subsequently merges with the horseshoe vortices creating strong PSHL that push the SSHL to the upper and lower corners of the suction side of the vane, as visible in Figure 11a. The optimized geometries *OPT-1* and *OPT-3* in Figure 11e,f behave similarly to each other, and in both cases, the PSHL almost disappears. This can be motivated by the smoother endwall diffusion and by the elongated vane profiles, which minimize the local variation of the cross-sectional area. As a consequence, the PSHL is attenuated and the SSHL results are bigger with respect to the baseline case. Figure 11b,c shows that the secondary structures in the optimized geometries are closer to the extremes in the span-wise direction, while the mid-span is almost not perturbated by recirculation zones.
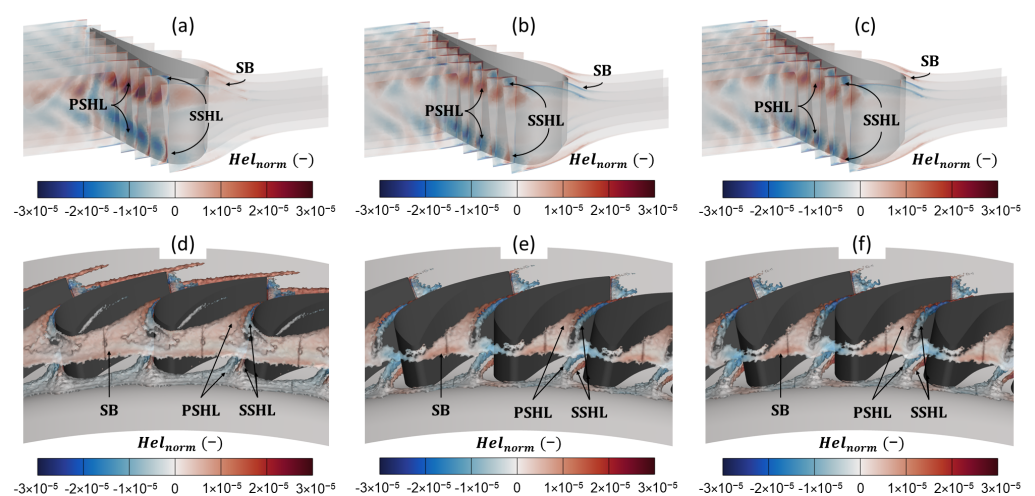

Figure 11. Helicity: (**a**) Baseline, (**b**) OPT-1, (**c**) OPT-3, (**d**) Baseline flow structures, (**e**) OPT-1 flow structures, and (**f**) OPT-3 flow structures.

These observations find consistency in the analysis of the outlet conditions, calculated on a plane extracted at an axial distance equal to the 30% of C downstream from the vane trailing edge. Here, the total pressure loss coefficient C_p in Figure 12 is plotted along the span-wise position. For the baseline design, the upper and lower vortices merge at the mid-span position creating a local peak in pressure loss. The optimized geometries *OPT-1* and *OPT-2* are instead characterized by the presence of two independent vortices at *Span* ≈ 0.2 and ≈ 0.7. Moreover, the exit flow yaw angle α_2 in Figure 12b shows a similar trend. The maximum flow deviation from $\alpha_{2,ref}$ occurs at the mid-span position $\Delta\alpha_{2,max} = 8.6°$ for the baseline geometry while coinciding for *OPT-1* and *OPT-2* with a peak of $\Delta\alpha_{2,max} = 5.2°$ located at the 70% of the span. The pitch angle δ is also reported in Figure 12c to provide insight into the intensity of the vertical component of the velocity, which can cause instability in the turbine. Fluctuations in δ are always lower than $1°$, and once again the optimized geometries present a smoother profile with smaller deviations. Finally, the Ma_2 profile in Figure 12d confirms that the outlet plane of the baseline geometry is strongly affected by secondary flows, which creates a local deceleration of the flow at the mid-span position. Both *OPT-1* and *OPT-3* are instead characterized by an almost uniform Ma_2 profile.
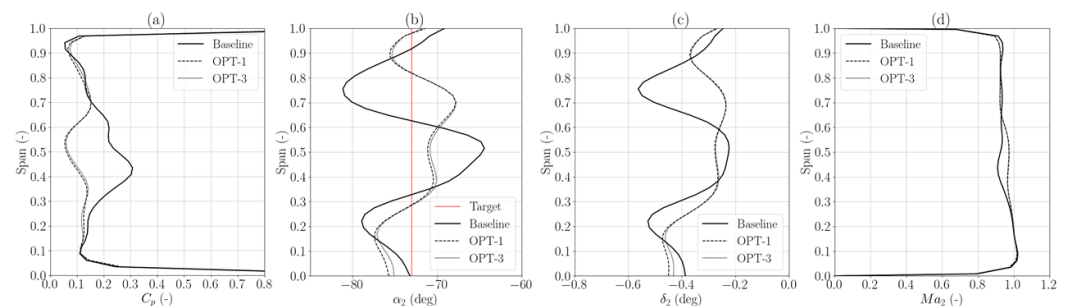

Figure 12. Outlet conditions: (**a**) pressure loss coefficient, (**b**) outlet yaw angle, (**c**) outlet pitch angle, and (**d**) outlet Mach number.

The inlet Mach number was found to be proportional to the overall mass flow rate, which changes according to the variation in the minimum throat area. This aspect can be quantified by introducing the contraction ratio CR, which is measured as the ratio between the inlet and the throat area. The baseline geometry in Figure 13a ingests the inlet flow rate at $Ma_1 \approx 0.5$ with a $CR = 1.34$. When Ma_1 is considered within the objective function (*OPT-3* in Figure 13c), the CR reduces up to 1.21 and $Ma_1 = 0.58$. In this context, solution *OPT-1* (Figure 13c) serves as a middle ground as it has $Ma_1 = 0.56$ and $CR = 1.23$. The isentropic Mach number in Figure 13d shows the occurrence of a shock at the 25% of the span and 75% of the axial chord for the baseline geometry, which is a source of aerodynamic losses. Moreover, the span-wise distribution of the load is not uniform. In the *OPT-1* case represented in Figure 13e instead, the load is evenly distributed over the span, as well as occurs for *OPT-3* in Figure 13e. The latter presents a stronger peak in terms of Ma_{is} at the 50% of the axial chord which coincides with the sudden curvature change in the suction side that produces a local over-acceleration. Eventually, *OPT-1* geometry in Figure 13e shows a better load distribution and this is conformal with the higher vane efficiency achieved by this geometry.

The pressure and velocity contour maps extracted at the mid-span are represented in Figures 14a–c and 15a–c. Both figures confirm the behavior showed by the Mach contours represented in Figure 13a–c, as the inlet flow initially decelerates, thus producing a local increase in the static pressure and then accelerates for the presence of the airfoil. Figure 15a–c represents a detailed view of the suction side in the $0.4 < C_{ax} < 0.7$ region, where the changes in the curvature affect the flow field behavior. In fact, the *OPT-3* solution (Figure 15c) is influenced by the strong curvature variation occurring close to $Z/C = 1$, which produces a local acceleration and a subsequent thickening of the boundary layer in the downstream region, where the pressure gradient becomes adverse as reported in

Figure 13f in the $0.5 < C_{ax} < 0.7$ region. This phenomenon is greatly mitigated in *OPT-1* (Figure 15b), and almost absent in the baseline case (Figure 15a).
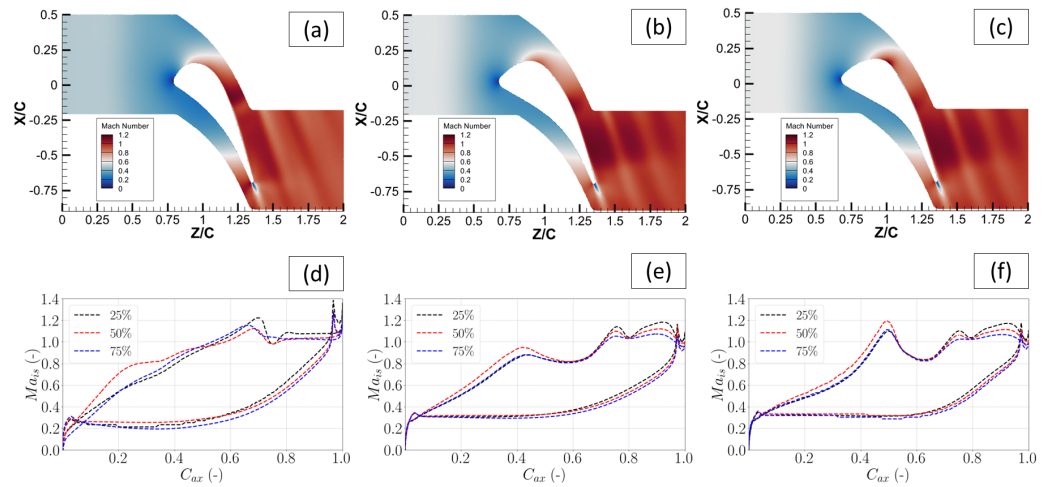

Figure 13. Mach contour: (**a**) baseline, (**b**) *OPT-1*, (**c**) *OPT-3*, (**d**) baseline isentropic Mach distribution, (**e**) *OPT-1* isentropic Mach distribution, and (**f**) *OPT-3* isentropic Mach distribution.
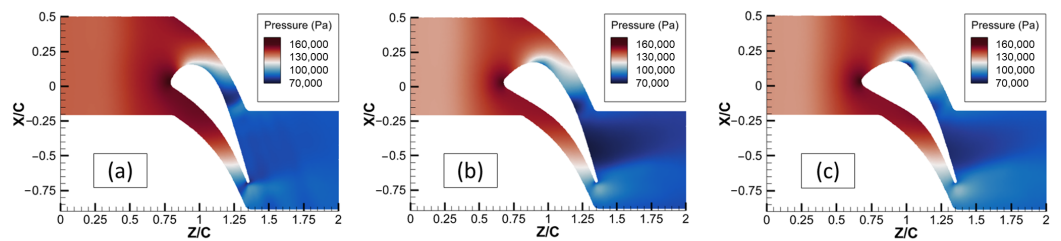

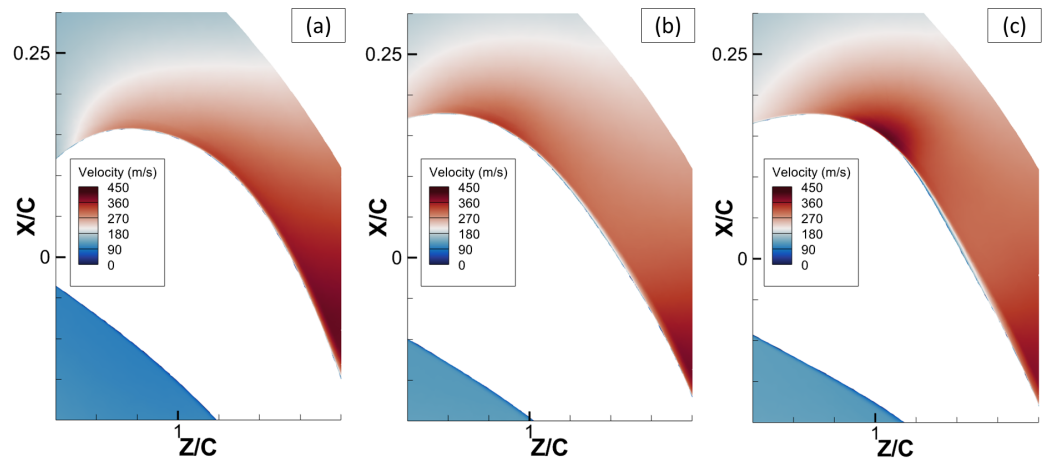**Figure 14.** Pressure contours at the mid-span: (**a**) baseline, (**b**) *OPT-1*, (**c**) *OPT-3*.

Figure 15. Velocity contours at the mid-span: (**a**) baseline, (**b**) *OPT-1*, (**c**) *OPT-3*.

7. Conclusions

The current framework focuses on the optimization of an HPT vane designed to ingest an inlet gas flow rate at $Ma = 0.6$. The vane and the diffusive endwall profiles were first parametrized with splines and then deformed by moving the position of the control points. This method was used to generate 885 different samples using an LHS approach that was tested through RANS CFD simulations. The aerodynamics of each sample was quantified with the vane efficiency. Furthermore, the Root Square Index Θ was introduced to quantify the impact of the new geometry on Ma_1, α_2, and η simultaneously. The overall dataset

was then used to train an ANN and an RF model aimed at predicting η and Θ as objective functions. Models were first tuned with a random search approach and then coupled with a genetic algorithm to search for optimal solutions.

The ANN model proved superior to the RF in generalizing predictions from test data, achieving $R^2_{test} = 0.98$ and $R^2_{test} = 0.99$ for the prediction of η and Θ, respectively, against $R^2_{test} = 0.89$ by RF. Moreover, the ANN model performance does not suffer from train-test split dependency, since cross-validated results are independent from the *k*-fold index. In the RF case, a significant dependency from data split can be noticed both during cross-validation and when the model is trained for increasing data size. Regarding the machine learning tools studied in this paper, it can be concluded that a properly tuned ANN does not encounter difficulties in managing the relation between the objective functions and the 18 input variables in their current range of variability. An increase in the design space or in the number of variables could be considered for a future study, in order to increase the complexity of the design and look elsewhere for a new optimum point. Since most of the computational effort of the present work is represented by data collection from CFD analysis, it is important to find a compromise between the predictive accuracy and the computational time. Analyzing the results from the ANN and the RF models, it is possible to conclude that the 885 samples are enough to achieve the desired level of prediction accuracy for the ANN model ($R^2 \approx 0.98$–0.99). Moreover, the R^2_{test} index for the RF model is strongly affected by the dataset size for $N < 600$. For $N > 600$ a slight increase in R^2_{test} is still observed, so it is not suggested to use a restricted dataset.

The current optimal geometries predicted by the models were thus simulated through CFD to verify the reliability of the predictions and to study the physics of the problem. The results demonstrated the excellent accuracy of the models, with the predictions matching the simulations within an error of approximately 1% for both ANN and RF. The major sources of aerodynamic losses can be identified in the generation of separation bubbles induced by the endwall diffusion angle, in the horseshoe structures created at the vane leading edge, and in the presence of shocks after the vane throat area. The optimized geometries in terms of η present strong mitigation of aerodynamic losses were obtained through a smoother endwall diffusion and an elongated vane shape. The combination of these geometrical features avoids sudden cross-sectional variation in the streamwise direction, preventing the flow separation from the diffusive endwalls. Similar considerations can be also extended to *OPT-3*, as the position of the leading edge and the shape of the endwalls coincide with *OPT-1*. However, *OPT-3* achieves $Ma_1 \approx 0.6$ by slightly increasing the vane throat area and thus the overall mass flow rate through the system.

Abbreviations

RDC	Rotating Detonation Combustor
HPT	High Pressure Turbine
LHS	Latin Hypercube Sampling
DOE	Design Of Experiments
ANN	Artificial Neural Network
RF	Random Forest
GA	Genetic Algorithm
OF	Objective function
ReLu	Rectified Linear Units
NAG	Nesterov accelerated gradient
CR	Contraction Ratio
SB	Separation Bubble
PSHL	Pressure Side Horseshoe Leg
SSHL	Suction Side Horseshoe Leg

Nomenclature

N	Number of samples
z	Summed weighted of the input
w	Weight
b	Bias
x	Input variable
L	Loss function
$\tilde{L}$	Regularized loss function
y	True value
$\hat{y}$	Model prediction
$\dot{m}$	Mass flow rate
T	Temperature
P	Pressure
h	enthalpy
U	mean velocity
u	velocity fluctuation
k	turbulent kinetic energy
S_M	momentum source term
S_E	energy source term
P_{kb}	buoyancy term in turbulent kinetic energy transport equation
$P_{\omega b}$	buoyancy term in turbulent dissipation rate transport equation
F_2	SST blending function
A	Area
Ma	Mach number
R	Gas constant
c_P	Specific heat at constant pressure
C_p	Total pressure loss coefficient
X	Lateral direction
Y	Vertical direction
Z	Axial direction
C	Chord of the vane
k	Cross validation fold index

Greek

η	Vane efficiency
Θ	Root Squared Index
α	Vane exit yaw angle
δ	Vane exit pitch angle
ρ	density
μ	dynamic viscosity
μ_t	turbulent viscosity
ν	kinematic viscosity
ω	turbulent dissipation rate

Λ	thermal conductivity
τ_{ij}	stress tensor
γ	Specific heat ratio
λ	L_2 regularization coefficient

Subscripts

train	Training dataset
test	Test dataset
val	Validation dataset
1	Inlet of the vane
2	Outlet of the vane
is	Isentropic
ax	Axial
th	Throat

References

1. Li, J.; Du, X.; Martins, J.R. Machine learning in aerodynamic shape optimization. *Prog. Aerosp. Sci.* **2022**, *134*, 100849. [CrossRef]
2. Queipo, N.V.; Haftka, R.T.; Shyy, W.; Goel, T.; Vaidyanathan, R.; Kevin Tucker, P. Surrogate-based analysis and optimization. *Prog. Aerosp. Sci.* **2005**, *41*, 1–28. [CrossRef]
3. Rai, M.M.; Madavan, N.K. Aerodynamic Design Using Neural Networks. *AIAA J.* **2000**, *38*, 173–182. [CrossRef]
4. Renganathan, S.A.; Maulik, R.; Ahuja, J. Enhanced data efficiency using deep neural networks and Gaussian processes for aerodynamic design optimization. *Aerosp. Sci. Technol.* **2021**, *111*, 106522. [CrossRef]
5. Li, J.; Cai, J.; Qu, K. Surrogate-based aerodynamic shape optimization with the active subspace method. *Struct. Multidiscip. Optim.* **2018**, *59*, 403–419. [CrossRef]
6. Zhang, C.; Janeway, M. Optimization of Turbine Blade Aerodynamic Designs Using CFD and Neural Network Models. *Int. J. Turbomach. Propuls. Power* **2022**, *7*, 20. [CrossRef]
7. Mengistu, T.; Ghaly, W. Aerodynamic optimization of turbomachinery blades using evolutionary methods and ANN-based surrogate models. *Optim. Eng.* **2007**, *9*, 239–255. [CrossRef]
8. Zhang, Y.; Sung, W.J.; Mavris, D. Application of Convolutional Neural Network to Predict Airfoil Lift Coefficient. *arXiv* **2018**, arXiv:1712.10082.
9. Du, Q.; Li, Y.; Yang, L.; Liu, T.; Zhang, D.; Xie, Y. Performance prediction and design optimization of turbine blade profile with deep learning method. *Energy* **2022**, *254*, 124351. [CrossRef]
10. Dasari, S.K.; Cheddad, A.; Andersson, P., Random Forest Surrogate Models to Support Design Space Exploration in Aerospace Use-Case. In *Artificial Intelligence Applications and Innovations*; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; pp. 532–544. [CrossRef]
11. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
12. Kramer, O.; Kramer, O. *Genetic Algorithms*; Springer: Berlin/Heidelberg, Germany, 2017.
13. Giannakoglou, K.C.; Papadimitriou, D.I., Adjoint Methods for Shape Optimization. In *Optimization and Computational Fluid Dynamics*; Springer: Berlin/Heidelberg, Germany, 2008; pp. 79–108. [CrossRef]
14. Jameson, A. *Aerodynamic Shape Optimization Using the Adjoint Method*; Lectures at the Von Karman Institute; Von Karman Institute: Brussels, Belgium, 2003.
15. Salvadori, S.; Insinna, M.; Martelli, F. Unsteady Flows and Component Interaction in Turbomachinery. *Int. J. Turbomach. Propuls. Power* **2024**, *9*, 15. [CrossRef]
16. Hishida, M.; Fujiwara, T.; Wolanski, P. Fundamentals of rotating detonations. *Shock Waves* **2009**, *19*, 1–10. [CrossRef]
17. Liu, Z.; Braun, J.; Paniagua, G. Integration of a transonic high-pressure turbine with a rotating detonation combustor and a diffuser. *Int. J. Turbo Jet-Engines* **2023**, *40*, 1–10. [CrossRef]
18. Grasa, S.; Paniagua, G. Design, Multi-Point Optimization, and Analysis of Diffusive Stator Vanes to Enable Turbine Integration into Rotating Detonation Engines. *J. Turbomach.* **2024**, *146*, 111002. [CrossRef]
19. Gallis, P.; Salvadori, S.; Misul, D.A. Numerical Analysis of a Flow Control System for High-Pressure Turbine Vanes Subject to Highly Oscillating Inflow Conditions. In *Turbo Expo: Power for Land, Sea, and Air, Volume 5: Cycle Innovations*; American Society of Mechanical Engineers: New York, NY, USA, 2024; Volume 87974, p. V005T06A021. [CrossRef]
20. Sieverding, C.; Arts, T.; De'nos, R.; Martelli, F. Investigation of the flow field downstream of a turbine trailing edge cooled nozzle guide vane. In Proceedings of the International Gas Turbine and Aeroengine Congress and Exposition, The Hague, The Netherlands, 13–16 June 1994; American Society of Mechanical Engineers: New York, NY, USA, 1994; Volume 1. [CrossRef]
21. Denos, R.; Sieverding, C.; Arts, T.; Brouckaert, J.; Paniagua, G.; Michelassi, V. Experimental investigation of the unsteady rotor aerodynamics of a transonic turbine stage. *Proc. Inst. Mech. Eng. Part A J. Power Energy* **1999**, *213*, 327–338.
22. McKay, M.D.; Beckman, R.J.; Conover, W.J. Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics* **1979**, *21*, 239–245. [CrossRef]
23. Roache, P.J. Verification of Codes and Calculations. *AIAA J.* **1998**, *36*, 696–702. [CrossRef]
24. Menter, F.R. Two-equation eddy-viscosity turbulence models for engineering applications. *AIAA J.* **1994**, *32*, 1598–1605.

25. ANSYS, Inc. *ANSYS CFX-Solver Theory Guide*; ANSYS, Inc.: Canonsburg, PA, USA, 2009.

26. Lau, M.M.; Hann Lim, K. Review of Adaptive Activation Function in Deep Neural Network. In Proceedings of the 2018 IEEE-EMBS Conference on Biomedical Engineering and Sciences (IECBES), Sarawak, Malaysia, 3–6 December 2018; IEEE: Piscataway, NJ, USA, 2018. [CrossRef]

27. Bejani, M.M.; Ghatee, M. A systematic review on overfitting control in shallow and deep neural networks. *Artif. Intell. Rev.* **2021**, *54*, 6391–6438. [CrossRef]

28. Dubey, A.K.; Jain, V., Comparative Study of Convolution Neural Network's Relu and Leaky-Relu Activation Functions. In *Applications of Computing, Automation and Wireless Systems in Electrical Engineering*; Springer: Singapore, 2019; pp. 873–880. [CrossRef]

29. Xu, B.; Huang, R.; Li, M. Revise Saturated Activation Functions. *arXiv* **2016**, arXiv:1602.05980.

30. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.

31. Dozat, T. Incorporating nesterov momentum into adam. In Proceedings of the 4th International Conference on Learning Representations, San Juan, Puerto Rico, 2–4 May 2016; pp. 1–4.

32. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.

33. Biau, G.; Scornet, E. A random forest guided tour. *TEST* **2016**, *25*, 197–227. [CrossRef]

34. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]

35. Lee, T.H.; Ullah, A.; Wang, R., Bootstrap Aggregating and Random Forest. In *Macroeconomic Forecasting in the Era of Big Data*; Springer International Publishing: Berlin/Heidelberg, Germany, 2019; pp. 389–429. [CrossRef]

36. Gad, A.F. Pygad: An intuitive genetic algorithm python library. *Multimed. Tools Appl.* **2023**, *83*, 58029–58042.