Real-Time Aerodynamic Performance Prediction of Multi Element Rear Wing for High-Speed Vehicles using ML and DL

[1]Kartikey Vishnu, [2]Devdoot Chatterjee, [3]Kartikey Gupta, [4]Raj Kumar Singh

[1,2,3,4]Mechanical Engineering Department, Delhi Technological University, New Delhi, 110042, India

Email: [1]kartikeyvishnu_2k19me123@dtu.ac.in, [2]devdootchatterjee_2k19me080@dtu.ac.in, [3]kartikeygupta_2k19me122@dtu.ac.in

Abstract:. The task of determining the aerodynamic properties of a multi-element airfoil has historically been a difficult, time-consuming process, and impractical for real-time applications. Machine learning (ML) and deep learning (DL) have shown promise in predicting aerodynamic properties. However, there has been limited research on using these techniques for multi-wing systems. In this study, we explore the use of ML algorithms to predict the aerodynamic performance of a multi-element wing, which has not been attempted in the literature. Training them on data generated using ANSYS Fluent, the models can predict these aerodynamic forces based on input parameters. Our approach has potential applications in optimizing the aerodynamic performance of high-speed vehicles, and may inspire further research in this area. The use of ensembling techniques further reduces computation time and power during offline stages. Thus reducing time taken for a cfd analysis from 10-15 minutes to 10-60 microseconds making it 1.0-6.0 * 10^8 times faster.

Index terms: *Airfoil, rear wing, Deep learning, multi-layer perceptron, autoencoders.*

I. INTRODUCTION

The aerodynamic design of a rear wing is critical to the performance of a high-speed vehicle, such as a race car. Several monographs, including those by Hucho [1], Katz [2], and McBeath [4], provide an excellent introduction to vehicle and race car aerodynamics. Rear wings are used to generate downforce and improve grip, which enhances stability and handling at high speeds. Multi-element wings, which consist of multiple airfoils stacked on top of each other, are becoming increasingly popular in motorsports due to their ability to generate higher levels of downforce while minimizing drag. For designing we referred to the works of S. Dhole et al. [5] and S. A. Patil et al. [6]. These articles discuss the design and optimization of the rear wing of a Formula SAE car using CFD simulations and a response surface methodology, along with a Taguchi-based optimization approach.

Traditionally, computational fluid dynamics (CFD) has been used to compute the aerodynamic properties such as the Coefficient of lift and the Coefficient of drag of vehicle components, such as airfoils and wings. However, this approach is computationally expensive and time-consuming, making it impractical for real-time applications.

In recent years, the application of machine learning (ML) and deep learning (DL) in the field of aerodynamics has shown promising results. Researchers have explored the potential of these techniques to predict aerodynamic properties such as lift and drag coefficients, pressure distribution, and boundary layer behavior for various aircraft and vehicle components.Several studies have focused on the prediction of aerodynamic properties of single-element airfoils using ML and DL. For instance, Huang et al. [7] developed a neural network model using back-propagation (BP) to estimate the lift and drag coefficients of a NACA63-215 airfoil under various flow conditions. Liu [8] constructed a radial basis function (RBF) neural network model to estimate the airfoil lift and drag coefficients within a specified parameter range.

While there has been some prior work on using ML and DL to predict the aerodynamic properties of airfoils, like Wallach, Santos, Mattos, et al. [9,10] predicted the lift and drag coefficients of NACA23012, the drag coefficients of a regional twinjet, and the drag coefficients of a wing-fuselage combination using MLP and functional link networks. Secco et al. [11] expanded on Wallach's work [9], using generic airfoils to create the wing-body combinations and assessing several artificial neural network (ANN) topologies, the use of these techniques for predicting the aerodynamic properties of multi-wing systems has not been explored extensively. Most of the work in this area has focused on using CFD simulations to compute aerodynamic properties and forces.

In this paper, we present a novel approach to predicting the aerodynamic performance of a multi-element wing using ML. In this study, we have explored the use of machine learning algorithms to predict the downforce and drag values of a multi-element rear wing at different angles of attack and velocities, a task that has not been previously attempted in the literature. To accomplish this, we performed Computational Fluid Dynamics (CFD) simulations of a three-element airfoil system using ANSYS Fluent software. We focused on the uppermost and center airfoils, as these are the most critical elements of the wing in terms of generating downforce.

Using the CFD simulations, we generated a dataset of downforce and drag values for a range of angles of attack and velocities. We then trained machine learning models to predict these aerodynamic forces based on the input parameters of angle of attack for the uppermost and the center airfoils, and velocity. We explored several different machine learning algorithms, including Random Forest, Support Vector Machine (SVM), Gradient Boosting, Adaboost, and Xgboost, and evaluated their performance in predicting the downforce and drag values.

One of the potential applications of this research is in the development of a dynamically changing rear wing system. By using the current velocity of the car, an on-board computer can determine which angle of attack is best suited for the current operating conditions without the need for performing additional CFD simulations. The machine learning models can be used to quickly compute the drag and downforce values for a range of angles in real-time, allowing the computer to adjust the angles of the rear wing airfoils to optimize the vehicle's aerodynamic performance. This application has the potential to significantly improve the performance of high-speed vehicles, and may be of interest to researchers and engineers working in the field of motorsports engineering. We believe that our approach has the potential to inspire further research in this area.

II. METHODOLOGY

Design and Dimensions of the Multi-Element Airfoil

The selection of the Eppler 423 airfoil for the rear wing of a sports car was a deliberate choice, based on its exceptional performance characteristics. The airfoil has a high lift-to-drag ratio (Eppler, 1990) [3], which is crucial for generating the necessary downforce to improve grip and stability at high speeds. The good stall characteristics of the airfoil are also advantageous, as they ensure that the wing remains effective even at high angles of attack.

To ensure optimal performance, the multi-element airfoil was designed with specific dimensions and angles. The bottom-most airfoil was chosen to have a chord length of 0.35m and an angle of 8° from the horizontal. The middle and uppermost airfoils were given chord lengths of 0.16m and 0.14m, respectively, with angles α_1 and α_2 that could be varied from 0-32° and 0-56° (Fig.1), respectively. These choices were based on suggested values of AOA of trailing elements presented by Simon McBeath (2006), which indicated that angles of 25–30 and 30–70 degrees for flap 1 and 2, respectively, were expected to produce the highest levels of downforce. The height of the multi-element system from the ground was kept at 0.5m, as this is a commonly used value in motorsports engineering. The

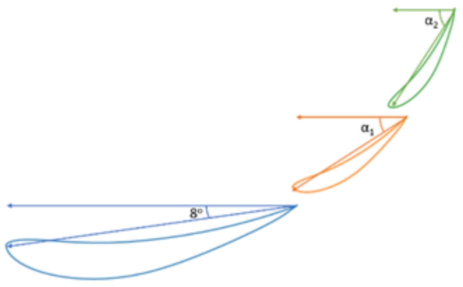


Fig.1 – Multi-element airfoil geometry (inverse E423)

combination of these design choices was expected to result in a significant increase in downforce and stability, which would be particularly beneficial for high-speed driving and racing applications.

Computational Domain and Geometry Setup

The design of the wings is two-dimensional to simplify the process and save computational time.

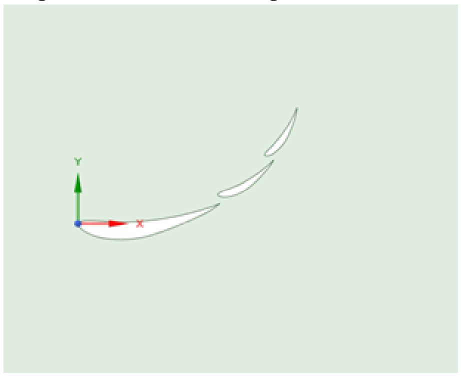
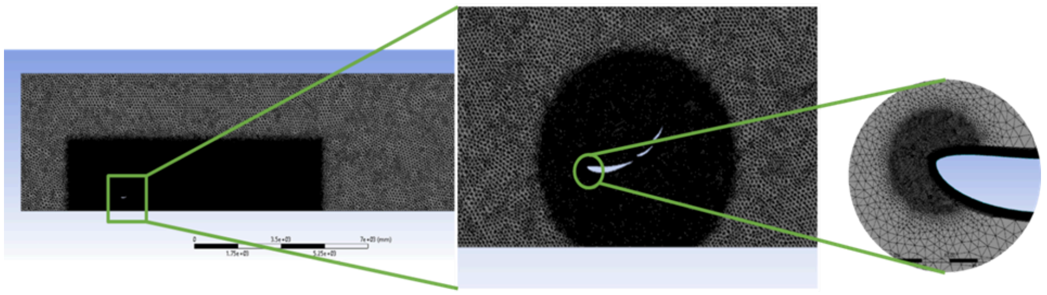


Fig.3 – Fluid domain in SpaceClaim (Close-up)

This approach neglects the formation of vortices at the wingtips, which can lead to an underestimation of lift and drag forces, as well as reduced aerodynamic efficiency. While it is true that vortices can eventually decay due to viscosity, in reality, they persist downstream of the wingtip. Nonetheless, a 2D analysis can still provide valuable insights and a fairly accurate



estimate of aerodynamic performance. A 3D analysis would significantly increase the number of variables to consider and require a higher level of computational power. The 2D geometry of the airfoil wing system was constructed, using Ansys Spaceclaim software. The angles, α_1 and α_2 were defined as input parameters. A rectangular fluid domain was created to perform the CFD and the system was kept at a height of 0.5m to account for ground effect.

Mesh Generation

For effective parameterization and automation of the CFD process in our project, a second-order, triangular, unstructured mesh was used to construct the fluid domain around the flow field. Though structured mesh is known to provide more accurate results, it requires a fixed and predetermined mesh that may not be suitable for automation.

Since we needed to perform multiple CFDs for different angles of the airfoil, it was necessary to use unstructured mesh. However, the quality of the mesh is critical to ensure accurate CFD results. Thus, the mesh was designed to have well-shaped cells with good aspect ratios and no overly stretched or distorted cells(Ferziger, J. H., and Perić, M., 2002) [28]. To capture the boundary layer and flow features accurately, multiple inflation layers were introduced and the mesh element size was reduced near the airfoil surface. This approach helped to improve the accuracy of the CFD simulations, despite the use of unstructured mesh.

Parameters	**Values**
Materials	Air (constant density of 1.225 kg/m^3)

Operating Parameters	Pressure = 101.36 Pa (or 1 atm))
	Temperature = 288.16 K

Boundary Conditions	Inlet: Magnitude of velocity = 14 m/s – 27 m/s Outlet: Pressure Outlet No-slip airfoil, ground
Viscosity	1.7894e-05 kg/(ms)
Ratio of Specific Heats	1.4
Initialization	Hybrid Initialization
Viscous Model	SST k-Omega

FLUENT Setup: Boundary Conditions and Solver Parameters

The flow was assumed to be constant, with absolute velocity formulation and planarity in two-dimensional space. The pressure-solver-based method was chosen because of its versatility in the range of Reynolds Numbers.

The SST k-Omega model was used to perform the CFD because it handles near-wall behaviour quite well, as cited by Kannan et al [14] and Darbandi et al [15], which is the case with multiple airfoils. The SST k-Omega model combines aspects of the k-Epsilon and k-Omega models to provide a more accurate and robust prediction of near-wall turbulence. It also resolves adverse pressure gradients far better than the k-Epsilon model, whose results are highly sensitive to the inlet Reynolds Number.

The inlet velocity was defined as an input parameter. The value of the velocity inlet was varied between 14 m/s to 27 m/s (both inclusive).

CFD Simulation Results

The solution for most wing systems was found to converge between 1e-7 to 1e-8 (SST k-Omega viscous model) and within 500 iterations of the same. The wall Y-plus was kept around 1 in our case.

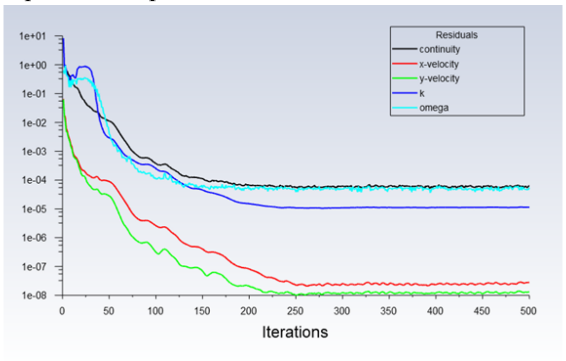


Fig.7 – Convergence of Residuals (α_1 = 32°, α_2 = 56°, velocity = 27 m/s)

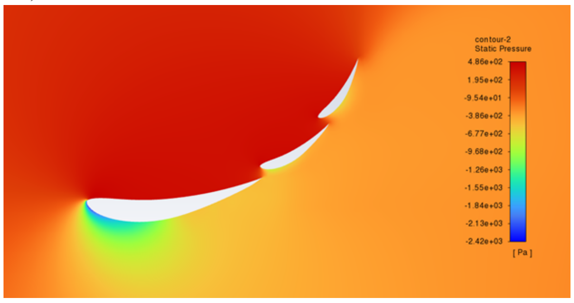


Fig.8 – Static Pressure distribution (α_1 = 32°, α_2 = 56°, velocity = 27 m/s)

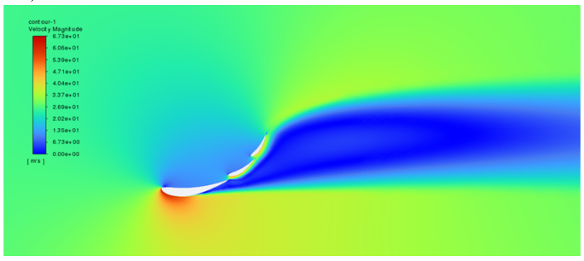


Fig.9 – Velocity distribution (α_1 = 32°, α_2 = 56°, velocity = 27 m/s)

Dataset generation

The values of downforce and drag were computed at different values of α_1 and α_2 ranging from 0-32° and 0-56° respectively, and for different values of velocity ranging from 14 m/s – 27 m/s using Ansys Fluent. Since the frontal area could not be parameterized, we used a Python script to compute the corresponding coefficient of drag (Cd) and coefficient of lift (Cl) values for each orientation and velocity. To do this, we first calculated the frontal area of the system in the direction of motion (A_x) and in the y-direction i.e., in upward direction (A_y) using a simple Python script that takes the airfoil coordinates of each wing and rotates them to the desired orientation to compute A_x and A_y. Then it computes Cl and Cd using the following formulas (J. Katz, 2006) [6]-

$$C_d = \frac{D}{(\rho V_\infty^2 A_x/2)}$$

$$C_l = \frac{L}{\left(\rho V_\infty^2 A_y/2\right)}$$

where:
ρ = Density of fluid, V_∞ = Free stream velocity.
D = Drag Force, L = Lift Force

Although we only considered a small range of velocity due to the limitations of the Formula students' vehicle, the method proposed in this study can be applied to a wide range of velocities.

Machine Learning

The application of Artificial Intelligence (AI) techniques have emerged as a promising approach for various domains. Machine learning is an automated learning paradigm that enables machines to learn and improve their performance by analyzing and identifying patterns in large datasets without being explicitly programmed. Ensemble methods first proposed by Breiman, Leo [19], combines multiple machine learning models to enhance the accuracy of predictions. Developing a machine learning model involves various critical phases, such as data preparation, feature selection, data pre-processing, model construction, and final prediction. In this study, a dataset was constructed by applying computational fluid dynamics to a multi-element airfoil, which involved collecting data on various parameters, including the Angle of mid and top airfoil, velocity, Cl, Cd, Lift force, and Drag force for each combination of angle and velocity. We looked into several ensemble approaches including Adaboost, Gradboost, and Xgboost, as well as multiple machine learning models like linear regression, support vector regression, and random forest. As detailed in section 2, for each combination of AoA(values for the mid and top airfoil sections) and velocity, the corresponding values of lift coefficient (Cl), drag coefficient (Cd), lift force, and drag force were recorded. Thus, each data point in the dataset included the following variables: Angle 1 (mid airfoil), Angle 2 (top airfoil), Velocity, Cl, Cd, Lift Force, and Drag Force. Ultimately, the finalized dataset consisted of over 1,100 observations and eight distinct parameters.

Analysis and Preprocessing

Statistical metrics

To assess the accuracy of continuous variable predictions, a variety of metrics such as RMSE, MAE, L2 Relative error as proposed in [21], and MSE are frequently utilized. The Mean Squared Error (MSE) is defined as the average of the squared differences between the actual and predicted values. The Root Mean Squared Error (RMSE) is calculated as the square root of the average of the squared differences between the actual and predicted values. Meanwhile, the Mean Absolute Error (MAE) is calculated as the average of the absolute differences between the actual and predicted values. Figure 2 displays the formulas corresponding to each of these metrics that were employed to evaluate the machine learning models in our study. All four metrics consistently indicate that lower values correspond to more reliable predictions.

$$L2\ Relative\ error = \sqrt{\frac{1}{n} * \sum_{i=1}^{n} \frac{\left(y_i - \hat{y}_i\right)^2}{y_i}}$$

$$RMSE\left(y, \hat{y}\right) = \sqrt{\left(\sum_{i}^{n} \frac{\left(y_i - \hat{y}_i\right)^2}{n}\right)}$$

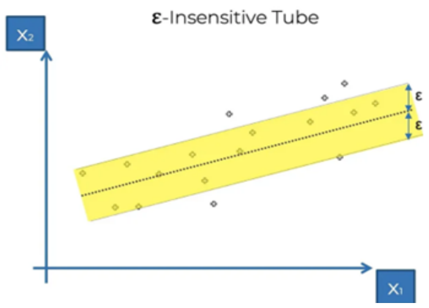
$$MSE\left(y, \hat{y}\right) = \sum_{i}^{n} \frac{\left(y_i - \hat{y}_i\right)^2}{n}$$

$$MAE\left(y, \hat{y}\right) = \sum_{i}^{n} \frac{\left(y_i - \hat{y}_i\right)}{n}$$

Where y_i is the true value whereas $\hat{y}_i$ is the predicted value. n is the total number of examples.

Model Architectures

1. SVM
Support Vector Regression (SVR) is a supervised learning algorithm used for regression tasks, first proposed by Vapnik et al. in 1996 [22]. It is based on Support Vector Machines (SVM). In the context of SVR, the objective is to identify a hyperplane that best fits the data points using support vectors (data points close to the hyperplane) while minimizing the error. SVR utilizes a kernel function to transform the input data into a higher dimensional space where it can be more easily separated. The most commonly used kernel functions in SVR are linear, polynomial, and radial basis function (RBF).
Figure 2 illustrates the concept of Linear support vector regression and the "ε-Insensitive Tube," which represents the error margin that the model is allowed to have.



In our investigation, we experimented with multiple kernels and found that the polynomial kernel worked particularly well for our problem statement.

$K(X_1, X_2) = (a + X_1^T X_2)^b$

The polynomial kernel is defined by increasing the power of the kernel to calculate the dot product. It produces n^d enlarged features with n original features and d degrees of polynomials, where d is the degree of the polynomial. We found degrees 9 and 7 to be the most suitable for the prediction of the coefficient of lift and drag, respectively.

2. Random Forest

In Random Forest Regression first proposed in [24], a large number of decision trees are created, each using a random subset of the training data and a random subset of the input features. Each tree makes a prediction for the target variable based on the input features, and the final prediction is calculated as the average of the predictions of all the trees.The number of trees that must be built before taking mean of prediction is a parameter that needs to be tuned. For our application, we found 5000 trees to be the most suitable for the prediction of the coefficient of lift and drag. The randomization of both the training data and input features helps to reduce overfitting and improve the generalization performance of the model. Random Forest Regression can handle a large number of input features and is robust to outliers and missing data.

3. Gradient boosting

The proposed approach first appeared in Friedman, Jerome H et al [25] describing a decision tree-based additive and sequential model that sequentially builds trees to transform weak learners into strong learners. This is achieved by increasing weights for the weak learners and decreasing weights for the strong learners. The number of trees to be constructed is a parameter that is optimized, with 10k trees being used for predicting the coefficient of lift and 15k for predicting the coefficient of drag. The model improves by learning from residual errors instead of updating the weights of individual data points, and each tree leverages the knowledge gained from the preceding tree's growth. The first step in gradient boosting is to build a base model to predict the observations in the training dataset using equation __.

$$F_0(x) = \arg_\gamma min \sum_{i=1}^{n} L(y_i, \gamma)$$

argmin means we have to find a predicted value/gamma for which the loss function is minimum. L is the loss function, for our application it is MSE.

In the next step, we will build a model on these pseudo residuals as shown in the equation below, and make predictions.

$$r_{im} = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]$$

Then we find the output values for each leaf of our decision tree and simply take the average of all the numbers in a leaf. The leaf's output value is the gamma value that minimizes the Loss function. Next, we update the predictions of the previous model. Input data point will go through this decision tree and the output gets will be multiplied with the learning rate and then added to the prediction of previous model. This is continued for the number of decision trees input to the model. We iterate through these steps until the loss function is negligible, finally we have our final output from the model.

4. Adaboost

The AdaBoost algorithm is designed to address classification errors by assigning greater weight to misclassified data points. As proposed by Freund, Yoav et al. [26], each data point in the dataset is given equal weight. A decision tree is then trained to determine its weighted error rate, which is a measure of the number of incorrect predictions made overall, and how each incorrect prediction is handled differently based on the weight of its underlying data point. The related mistake is given more significance when calculating the weighted error rate, with the weight being dependent on the weight of the data point. The ensemble weight of a decision tree is determined using a learning rate* log((1 — e)/e), and it signifies the decision power of the tree and its ability to reduce the weighted error rate. The algorithm subsequently updates the weights of the incorrectly classified points, and these steps are repeated iteratively until the predetermined number of decision trees is reached. Finally, the algorithm makes the final prediction.

5. Extreme Gradient boosting or Xgboost

XGBoost is a distributed gradient boosting toolkit that has been specifically optimized for fast and scalable machine learning model training. The ensemble learning technique employed by XGBoost combines the predictions of multiple weak models to obtain a stronger prediction. Chen et al. in [27] proposed "XGBoost" is short for "Extreme Gradient Boosting,". This approach refers to the sequential generation of decision trees. Prior to being incorporated into the

Coefficient of Lift (Cl) (Actual)	Coefficient of Lift (Cl) (Predicted)	Error in prediction (Cl)	Coefficient of Drag (Actual)
3.629236	3.629589	0.000353	0.269406
2.757205	2.757684	0.000478	0.312640
4.156257	4.1554112	0.000846	0.287968
1.712937	1.7119294	0.001008	0.247702
2.770037	2.771303	0.001266	0.334972

Table 3: Ground Truth Cl, Cd values, predicted Cl, Cd values and error in their prediction by Xgboost for various angle of attacks and velocity

decision tree that predicts outcomes, each independent variable is assigned a weight. Variables that the tree incorrectly predicted are assigned greater weight before being included in the second decision tree. The number of trees to be used is a hyperparameter that was tuned to achieve optimal performance, with values of 10,000 and 15,000 being found to be optimal for predicting the coefficients of lift and drag, respectively. These distinct classifiers/predictors are subsequently combined to produce a robust and highly accurate model.

Maths behind it:

Model architectures	**MSE Score**	**RMSE Score**	**MAE Score**	**L2 relative error**	**Time taken**
SVR	0.08202	0.28491	0.138614	0.06800	5.31x10^(-5)
Random forest	0.00803	0.07718	**0.017566**	0.00319	3.62x10^(-3)
Gradient boost	0.00940	0.08432	0.020209	0.00442	5.83x10^(-5)
Adaboost	0.07253	0.22661	0.179105	0.03367	3.54x10^(-5)
Xgboost	**0.00745**	**0.07375**	0.0215	**0.00282**	5.01x10^(-4)

The ultimate score is then calculated by adding the prediction scores of each individual decision tree.

$$\hat{y}_i = \sum_{k=1}^{K} \boldsymbol{f}_k(x_i),\ \boldsymbol{f}_k \in \boldsymbol{F} \text{ (x\_i)}$$

Table 2: Metric score corressponding to each ML algorithm used.

where, K is the number of trees, f is the functional space of F, F is the set of possible CARTs. The objective function for the above model is given by:

$$obj(\theta) = \sum_{i=1}^{n} \ell\left(y_i, \hat{y}_i\right) + \sum_{k=1}^{K} \Omega\left(\boldsymbol{f}_k\right)$$

where the loss function comes first and the regularisation parameter comes second. Thus, rather of learning the tree all at once, which makes optimisation more difficult, we use the additive strategy, minimise the loss from what we have learned, and add a new tree. Here we define the model as

$$\boldsymbol{f}_t(x) = w_{q(x)},\ w \in R^T,\ q : R^d \rightarrow \{1, 2, \ldots\ldots T\}$$

w is the vector of scores on leaves of tree, q is the function assigning each data point to the corresponding leaf, and T is the number of leaves. The regularization term is then defined by:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

where gamma is pruning parameter or the least information gain to perform split. We attempt to optimise one level of the tree at a time by dividing it into two in order to gauge how effective the tree is.

RESULTS AND DISCUSSION

All machine learning algorithms listed in Table 2 were employed to predict the coefficients of lift and drag (Cl and Cd) using the velocity and angle of each airfoil as inputs. The proposed architectures were evaluated based on the metrics discussed in Section 2, and the results are presented in Table 2. These results demonstrate that the random forest, gradient boost, and extreme gradient boosting (XGBoost) algorithms perform exceptionally well on our given dataset, yielding similar scores on all the evaluated metrics. These algorithms predict the coefficients of lift and drag with high accuracy, with L2 relative errors as low as 0.002-0.003 and mean squared error (MSE) scores as low as 0.0075-0.0095. Among these algorithms, XGBoost outperforms the others, achieving the lowest score on almost all the metrics used for evaluation, while gradient boosting has scores similar to those of XGBoost and random forest but requires less time to predict the coefficients of lift and drag. Our results indicate that computational fluid dynamic (CFD) analysis for each combination of multi-element airfoil requires 10-15 minutes, whereas the machine learning algorithm can complete an analysis in the order of 10-60 microseconds. This suggests that our proposed technique is 1.0 to 6.0 * 10^8 times faster than existing CFD analysis. Notably, the proposed machine learning algorithms are sensitive to hyperparameters, and significant effort was spent tuning them. In addition, Table 2 reports the actual and predicted values for the coefficients of lift and drag obtained from the XGBoost technique, along with the differences between the predicted and actual values. These results clearly demonstrate that our machine learning model yields not only fast but also highly accurate predictions.

CONCLUSION

The present study demonstrates the application of a machine learning algorithm as a computational tool for performing CFD analysis on a multi element airfoil using only velocity and angle as input parameters. Results indicate that the use of machine learning models allows for faster prediction of coefficients of lift and drag compared to standard CFD analysis while maintaining a high degree of accuracy. It has been observed that an analysis that takes 15-20 minutes can be done in 10-60 microseconds making it 1.0-6.0 * 10^8 times faster. Our methodology can be extended to predict the coefficients of lift and drag for different combinations of velocities and orientations of a multi-element airfoil. This approach can contribute to many applications. Such as it enables us to identify suitable orientations and angle combinations for each airfoil, based on specific requirements or use cases. For instance, maximizing Cl can provide the most suitable orientation for applications requiring high lift or downforce. Maximizing Cl/Cd ratio can provide the best orientation for high lift and low drag. Our study lays the foundation for analyzing multi-element airfoils using machine learning, and future research efforts could explore the use of deep learning and reinforcement learning for multi-element airfoil analysis, while addressing the challenge of generating large datasets for improved generalization. While our current study was limited to analyzing a single type of airfoil, our proposed approach can be extended to generalize on various types and series of airfoils. Overall, our approach presents a promising and computationally efficient way to perform CFD analysis on multi-element airfoils using machine learning, paving the way for further advancements in the field.

REFERENCES

1. Hucho, W., The aerodynamics of road vehicles, Butterworths Publishers, London, 1965.
2. Katz, J., Race Car Aerodynamics, Bentley Publishers, USA, 2006.
3. Eppler, R. (1990). Airfoil Data. In: Airfoil Design and Data. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-02646-5_6
4. McBeath, S., Competition Car Downforce, Haynes Publishers, Somerset, 1998.
5. S. D. Dhole and S. B. Joshi. "Design and Optimization of Rear Wing of Formula SAE Car"
6. S. A. Patil et al. "Design and Analysis of Rear Wing of Formula SAE Car for Optimum Aerodynamic Performance" http://dx.doi.org/10.11835/j.issn.1000-582X.2017.10.005
7. Jihong, H.; Su, H.; Zhao, X. Aerodynamic coefficient prediction of airfoil using BP neural network. Adv. Aeronaut. Sci. Eng. 2010, 1, 36–39. https://doi.org/10.48550/arXiv.2109.12139
8. Xin, L. Simulation of Airfoil Plunging Aerodynamic Parameter Prediction Based on Neural Network. Comput. Simul. 2015, 12, 18. http://dx.doi.org/10.3390/app12105194
9. Wallach, R.; Mattos, B.; Girardi, R.; Curvo, M. Aerodynamic coefficient prediction of transport aircraft using neural networks. In Proceedings of the 44th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA, 9–12 January 2006; p. 658 https://doi.org/10.1108/AEAT-05-2014-0069.
10. Santos, M.; Mattos, B.; Girardi, R.(2021) Aerodynamic coefficient prediction of airfoils using neural networks. In Proceedings of the 46th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, USA, 7–10 January 2008; p. 887. https://doi.org/10.48550/arXiv.2109.12149
11. Secco, N.R.; Mattos, B.S. Artificial neural networks to predict aerodynamic coefficients of transport airplanes. Aircr. Eng. Aerosp. Technol. 2017, 89, 211–230. http://dx.doi.org/10.1108/AEAT-05-2014-0069
12. Hai Chen, Lei He, Weiqi Qian and Song Wang(2020) Multiple Aerodynamic Coefficient Prediction of Airfoils Using a Convolutional Neural Network. https://doi.org/10.3390/sym12040544
13. Viren Chiplunkar, Rahul Gujar, Abhir Adiverekar, Rahul Kulkarni(2022). Computational fluid dynamics analysis for an active rear-wing design to improve cornering speed for a high-performancecar https://doi.org/10.1016/j.matpr.2022.12.040
14. Kannan, B.T., Karthikeyan, S., Senthilkumar Sundararaj (2017). Comparison of Turbulence Models in Simulating Axisymmetric Jet Flow. Innovative Design and Development Practices in Aerospace and Automotive Engineering. Lecture Notes in Mechanical Engineering. Springer, Singapore. https://doi.org/10.1007/978-981-10-1771-1_43
15. Darbandi, M., Zakyani, M., & Schneider, G. (2005). Evaluation of Different k-omega and k-epsilon Turbulence Models in a New Curvilinear Formulation. 17th AIAA Computational Fluid Dynamics Conference. https://doi.org/10.2514/6.2005-5101
16. Xinyu Hui, Junqiang Bai, Hui Wang, Yang Zhang (2022). Fast pressure distribution prediction of airfoils using deep learning. https://doi.org/10.1016/j.ast.2020.105949
17. Xuxiang Sun, Wenbo Cao, Yilang Liu, Linyang Zhu, Weiwei Zhang (2014). High Reynolds number airfoil turbulence modeling method based on machine learning technique https://doi.org/10.1016/j.compfluid.2021.105298
18. Achour, G., Sung, W. J., Pinon-Fischer, O. J., &amp; Mavris, D. N. (2020). Development of a conditional generative adversarial network for airfoil shape optimization. AIAA Scitech 2020 Forum. https://doi.org/10.2514/6.2020-2261.
19. Breiman, Leo. "Bagging predictors." Machine Learning, vol. 24, no. 2, 1996, pp. 123-140.
20. Hébert, P. M., and R. T. Sikora. "A Note on Normalization of Multidimensional Min-Max Normalization." Pattern Recognition, vol. 6, no. 3, 1974, pp. 191-193.
21. Bishop, C. M. (1994). Novelty detection and neural network validation. IEE Proceedings - Vision, Image and Signal Processing, 141(6), 307-315.
22. Drucker, H., Burges, C. J., Kaufman, L., Smola, A., & Vapnik, V. (1997). Support vector regression machines. Advances in neural information processing systems, 155-161.
23. Galton, F. (1886). Regression towards mediocrity in hereditary stature. Journal of the Anthropological Institute of Great Britain and Ireland, 15, 246-263.
24. Breiman, Leo. "Random Forests." Machine Learning, vol. 45, no. 1, 2001, pp. 5-32.
25. Friedman, Jerome H. "Greedy Function Approximation: A Gradient Boosting Machine." Annals of Statistics, vol. 29, no. 5, 1999, pp. 1189-1232.
26. Freund, Yoav, and Robert E. Schapire. "A Short Introduction to Boosting." Journal of Japanese Society for Artificial Intelligence, vol. 14, no. 5, 1995, pp. 771-780.
27. Chen, Tianqi and Carlos Guestrin. "XGBoost: A Scalable Tree Boosting System." Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 785-794.
28. Ferziger, Joel H., and Milovan Peric. Computational Methods for Fluid Dynamics. Springer, 2002.