Reconstructing Aerodynamic Flow Fields using Machine Learning

By M. Ishak, Supervised by Dr. S. Symon

UNIVERSITY OF Southampton

mi1g19@soton.ac.uk
ID:31032281

Objectives

To reconstruct aerodynamic flow fields, these objectives must be met:

1. Analyse the data and use this information to select an appropriate method of machine learning.
2. Create a machine learning model to represent how the data is mapped.
3. Compute, return and evaluate the predictions made.

Introduction

Refining airflow data through machine learning is a very important concept in aerodynamics as it allows for a practitioner of computational fluid dynamics to elucidate the airflow physics from the numerical data. Before machine learning, engineers would use their own understanding of flow to extrapolate the flow field, meaning it was not accurate. Machine learning removes the problem of random errors, improving resolution.

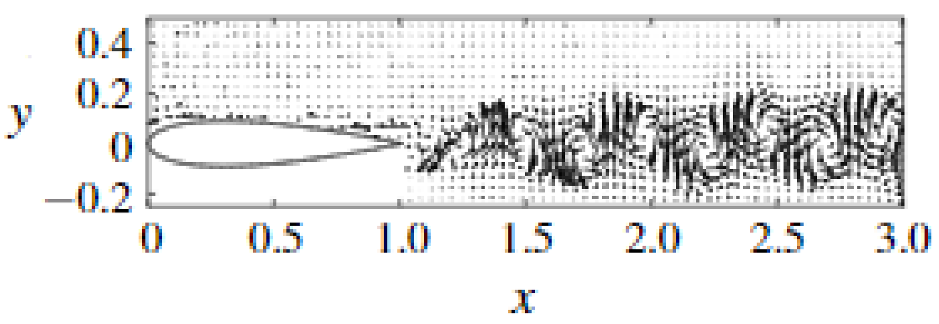

Figure 1: PIV vector plots of the flow field about a NACA0018 airfoil at an angle of attack of $\alpha = \mathbf{0}\,\mathbf{deg}$ (Symon et al., 2019).

This project cleans the data taken from the PIV experiment in Figure 1 of any outliers, looks for correlation between the input and output data-sets and creates a model that will provide an accurate extrapolation of this field.

Methodology

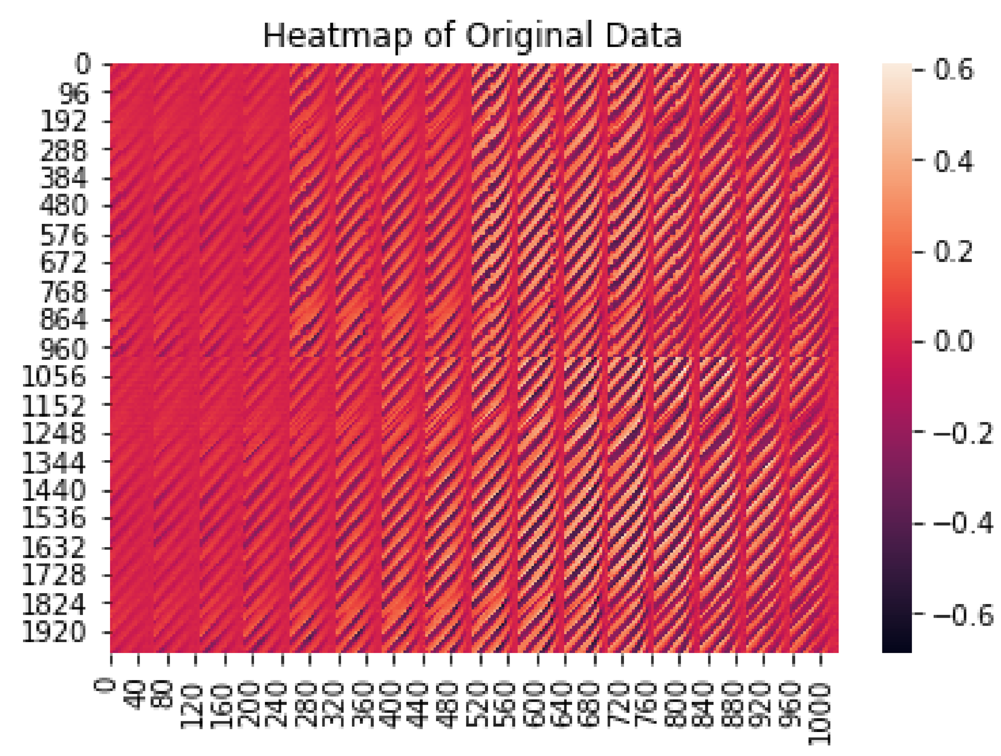

Figure 2: Heatmap of the input and output cross-product data.

A correlation matrix heat map (Figure 2) was plotted to show the strong link between the input and output data. Owing to pressure decay in airflow, a linear regression model was not optimal as decay is non-linear, meaning a sequential model was used (see model below).

$$\begin{cases} X_n^i = X_n, & n = 1, \dots, N^i, \\ X_n^{h1} = \vartheta^{h1}\left(\sum_{m=1}^{N^i} W_{n,m}^{h1} Y_m^i + B_n^{h1}\right), & n = 1, \dots, N^{h1}, \\ X_n^{h2} = \vartheta^{h2}\left(\sum_{m=1}^{N^{h1}} W_{n,m}^{h2} Y_m^{h1} + B_n^{h2}\right), & n = 1, \dots, N^{h2}, \\ X_n^{o} = \vartheta^{ho}\left(\sum_{m=1}^{N^{h2}} W_{n,m}^{o} Y_m^{h2} + B_n^{o}\right), & n = 1, \dots, N^{o}, \\ \hat{X}_n = X_n^o, & n = 1, \dots, N^o, \end{cases}$$

where $N^i = \mathbf{1600}$, $N^{h1} = N^{h2} = \mathbf{3200}$, $N^o = \mathbf{1024}$ and represents the numbers of dense neurons in each layer. In Figure 3, W are the weights and B are the biases, where weights are the parameters within the neural network.

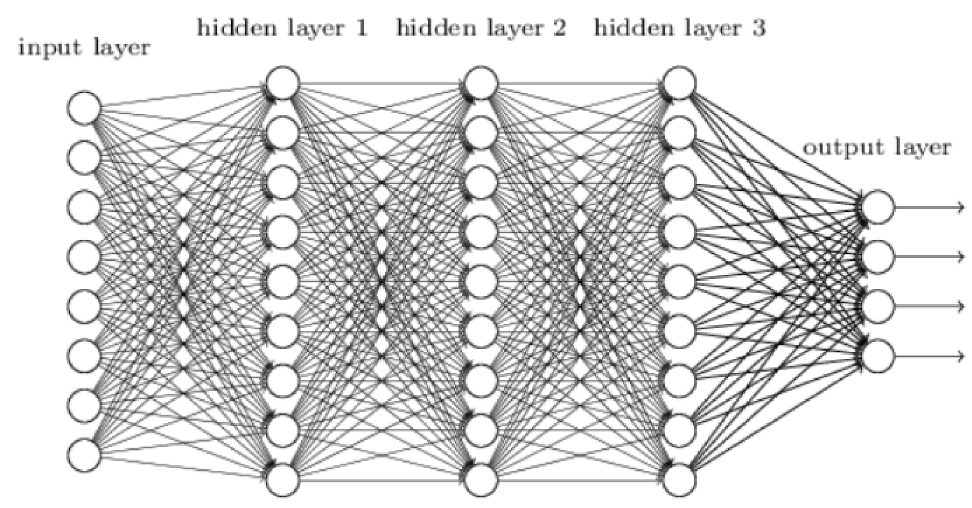

Figure 3: Schematic for the deep neural network used in this method (Goodfellow et al., 2017).

Results and Discussion

Layer (type)	Output Shape	Number of Parameters
First (Dense)	(None, 1600)	1640000
Second (Dense)	(None, 3200)	5123200
Third (Dense)	(None, 3200)	10243200
Fourth (Dense)	(None, 1024)	3277824
Total Parameters		20,284,224
Trainable Parameters		20,284,224
Non-trainable Parameters		0

Table 1: Summary of the resulting sequential deep neural network model created.

This model uses the equations in the previous section to learn the mapping between the input and output data-sets. This mapping is then translated onto the test data-set to extrapolate the next 20,284,224 parameters (Table 1). A comparison between the predicted, input and validation data-sets is shown in Figure 4.

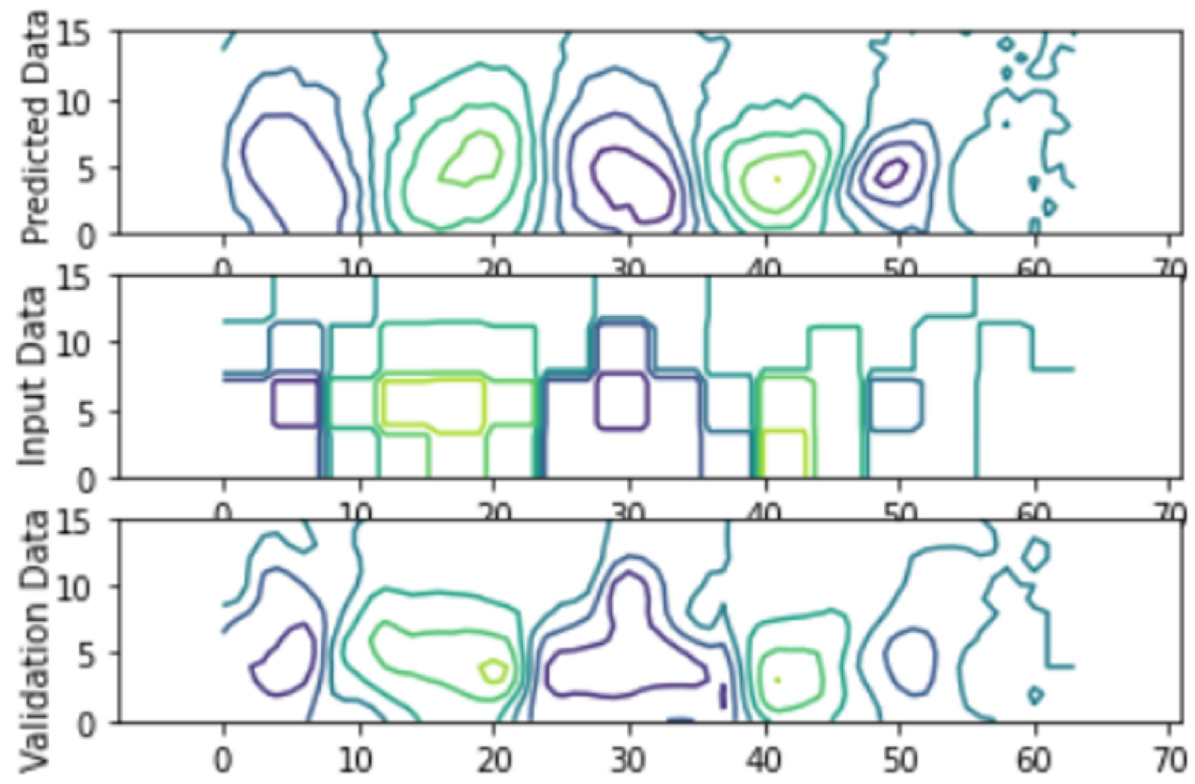

Figure 4: Contour plots showing the predicted, input and validation data-sets respectively from top to bottom.

Using 100 epochs, a batch size of 100, a test split of 0.2, as well as the ADAM optimizer (with a learning rate of $\boldsymbol{\lambda} = \mathbf{1} \times \mathbf{10^{-4}}$) and the Sigmoid activation function; the predicted and validation plots (Figure 4 4) indicate an accurate prediction. This configuration also showed that:

- The high computational complexity meant that the code was expensive.
- Predictive performance trades off with interpretability. A suitable optimizer and activation function must be used for representative data.
- Increasing the training split meant that there was a higher input but with a reduced predicted performance.
- Increasing the number of epochs improved the predictive performance.
- Coefficient of Determination (R^2) indicated overfitting in the data.

Conclusions and Considerations for the Future

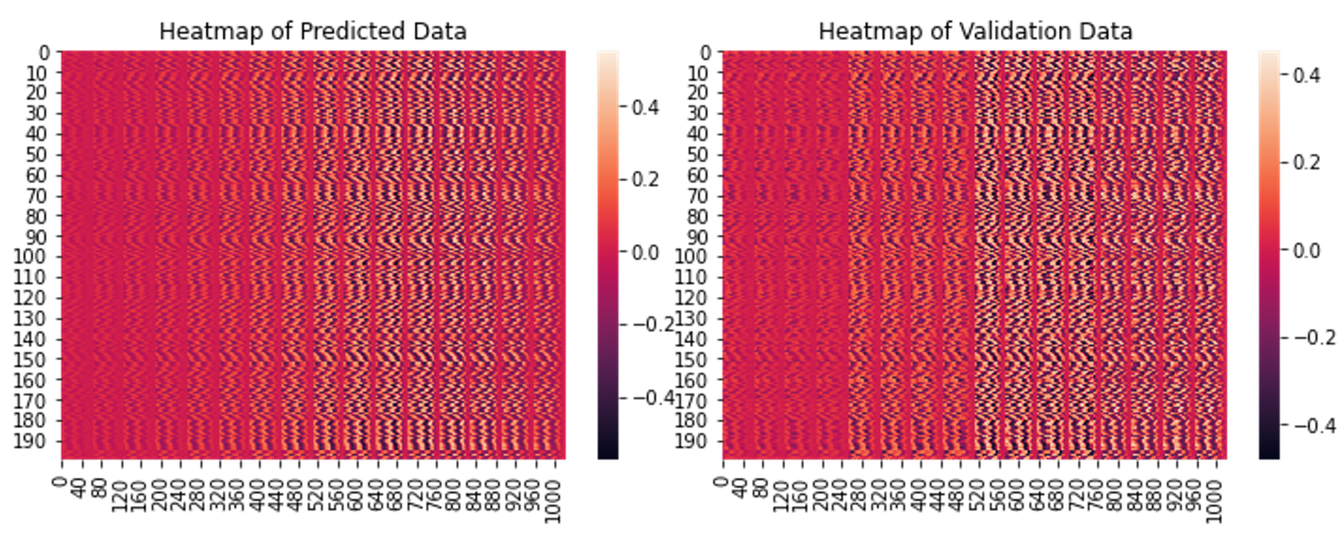

Figure 5: Heatmaps of the Predicted and Validation Data

Reconstructing flow fields using machine learning has shown to be useful, accurate (see Figure 5) and representative albeit computationally taxing. To reduce overfitting and computational complexity, a convolutional neural network should be considered for future work.

References

Goodfellow I., Bengio Y., Courville A. (2017). *Deep learning*. MIT Press.

Symon S., Sipp D., McKeon B. J. (2019). A tale of two airfoils: resolvent-based modelling of an oscillator versus an amplifier from an experimental mean. *Journal of Fluid Mechanics, 881*, 51-83. doi: 10.1017/jfm.2019.747