EURASIAN JOURNAL OF MATHEMATICAL AND COMPUTER APPLICATIONS
ISSN 2306–6172
Volume 13, Issue 4 (2025) 179 – 192

STUDY OF NUMERICAL MODELS OF TURBULENCE, BASED ON MACHINE LEARNING METHODS FOR SCALAR FLUX IN HEAT-RELEASED ASSEMBLIES

Tokarev M.P. [1], **Lukyanov A.A.**, **Yakovenko S.N.**

Abstract In this paper, an analysis of turbulence models created using machine learning methods is performed for a wide range of fluid flow parameters at low Prandtl numbers in a channel with complex shape. Three different modern machine learning methods are used for the analysis. A developed turbulent flow in the peripheral channel fragment with three round longitudinal rods is considered for Reynolds number Re = 4270 at low Prandtl numbers Pr = 0.025, 0.05, 0.1, 0.2, 0.4, 0.6, 0.71 encountered in heat-released elements where liquid metals and gas mixtures are used as coolant. The importance of input features of turbulent scalar flux models is analyzed for the entire data set using the permutation feature importance assessment and a method based on Shapley values. The importance of using the local Reynolds number invariant for accurate prediction of the transverse turbulent scalar flux components is demonstrated. It is found that the mutual correlation of some invariants within groups of the selected 15 input features generally preserves the significance level of the most important invariants within each of the groups for the universally interpretable machine learning model for the turbulent scalar flux modeling. Using the permutation method, it was found that the most important basis tensor for estimating the transverse components of the turbulent scalar flux in the tensor basis neural network model is the tensor obtained by the inner product of the deformation tensor with itself S^2, and the least important is the tensor $SR + RS$.

Keywords: turbulence models, scalar turbulent flux approximation, small Prandtl numbers, machine learning, RANS-ML.

AMS Mathematics Subject Classification: 76-10, 76D05, 76F99.

DOI: 10.32523/2306-6172-2025-13-4-179-192

1 Introduction

Turbulent flows are complex phenomena characterized by irregular and chaotic fluctuations in velocity, pressure, and other properties of a fluid. These fluctuations occur for a wide range of length and time scales, making it difficult to accurately predict and model turbulent flows. Therefore, there is an urgent need in industry to develop efficient and reliable numerical simulation methods based on various solutions of the Navier–Stokes equations for velocity and pressure fields, as well as those for the equation for a scalar field (temperature or concentration).

Direct numerical simulation (DNS) is the most accurate method, but it requires very high resolution and is computationally expensive, as is the large-eddy simulation (LES)

[1]Corresponding Author.

method, which explicitly resolves large- and medium-scale turbulent structures. For example, simulating the flow around a car or an airplane using DNS and LES will only be possible in 2080 and 2045, respectively [1]. Therefore, Reynolds-averaged Navier–Stokes (RANS) models are the most attractive approach in engineering applications due to their computational efficiency now and in the foreseeable future [2].

However, due to the low accuracy of the RANS approach, the task to improve existing turbulence models remains relevant, including the model development with the help of machine learning (ML) algorithms [3]. For this purpose, available high-fidelity data sets for canonical fluid flows obtained from measurements or numerical simulations using the eddy-resolving DNS and LES methods are used.

Machine learning methods have been introduced into computational fluid dynamics because they can provide additional benefits in analyzing the large number of available numerical and experimental data sets [4]. Some such ML approaches include the multidimensional gene expression programming (MGEP) based on symbolic regression [5] and gradient descent methods to optimize the weights of neural network architectures to estimate adjustments to baseline models. Several neural network architectures have been developed for calibrating turbulence models, namely: TBNN [6], UIML [7], EVNN with apNN [8], CNN with SVM [9]. Some of the trained models [8, 9] can be compared to a system called a black-box model due to their complex and intricate structure and computations that are difficult to interpret. In such a model, the resulting turbulence closure terms are calculated from a redundant number of input flow features through a sequence of connected neurons, where the number of initial connections is greater than the number of ultimately significant output predictions. Other models are partially explainable [6, 7] due to the special design of the computational graph of the models. This special form arises from physical restrictions on the Galilean invariance of the closure terms [10]. On the other hand, improving the explainability of a model will make it simpler, and the model may ultimately lose its ability to make accurate predictions [11].

Another approach to explainability analysis is to use model-independent methods, which are local approaches that can explain even individual predictions, such as the LIME method [12] based on local surrogate models and SHAP [13] based on Shapley values. One of the difficulties in interpreting black box models is the correlation of different features [14], which can lead to unrealistic feature importance values when interpreting methods.

This paper focuses on the explainability and interpretation of the above-mentioned models obtained using ML methods on the data for a peripheral fragment of a channel with three longitudinal rods. Such configurations are encountered in rod bundles in nuclear reactors, where liquid metals and gas mixtures are used as coolants. The content of the paper below is structured as follows. Section 2 describes the calculated flow geometry configuration and describes the numerical experiments performed using the DNS method to obtain high-fidelity data that serve as a reference for improving the RANS approximations. Section 3 describes the machine learning approaches that are used to improve the baseline RANS turbulence models used to predict the scalar field. Section 4 presents the methods for analyzing the importance of input features for the machine learning models obtained. Section 5 summarizes the main results of

the model interpretation study.

2 DNS framework

First, numerical studies were carried out using DNS for a model of a peripheral fragment of a channel with a bundle of rods with a Reynolds number of $Re = U_b D_h/\nu = 4270$, where U_b is the bulk velocity, ν is the kinematic molecular viscosity and D_h is the hydraulic diameter. The preliminary results of velocity field predictions using DNS with a comparison of flow behavior in the peripheral and internal fragments of the channel can be found in [15]. The Nek5000 computational code is used to solve the Navier–Stokes equations of an incompressible fluid using a third-order time integration scheme and spatial discretization based on the spectral element method [16]. In DNS calculations, a three-dimensional problem is considered, as shown in Fig. 1, with no-slip conditions on the walls limiting the flow in sections (y, z). Periodic boundary conditions are applied along the x axis, which allows subsequent averaging of data both over time t and along the longitudinal direction x, providing two-dimensional fields of mean velocity $\overline{u}_i$, mean pressure $\overline{p}$, Reynolds stresses $\overline{u_i' u_j'}$ in the (y, z) plane. In the simulation a total of $N_{tot} = 17.92 \times 10^6$ grid points are used, which meets the DNS criteria for the Reynolds number under consideration $Re = U_b D_h/\nu = 4270$. The number of points in streamwise direction was $N_x = 400$, in crosswise directions $N_y \approx 500$ and $N_z \approx 90$ correspondingly. The number of spectral elements with one hundred of points in each element is shown in Fig. 1 for the cross section yz in the upper half of the DNS computation domain. The integration period is taken to be $T = 100 D_h/U_b$, which is sufficient to collect stable statistics. A posteriori, the Kolmogorov scale was estimated from the obtained DNS data using the following expression: $\eta = (\nu^3/\epsilon)^{0.25}$, where ν is the kinematic viscosity, ϵ is the viscous rate of dissipation of turbulent kinetic energy. The Kolmogorov microscale is $\eta = 3.6 \times 10^{-5}$ m, which in order of magnitude corresponds to the minimum cell size $\lambda = 6.3 \times 10^{-5}$ m for DNS calculation.

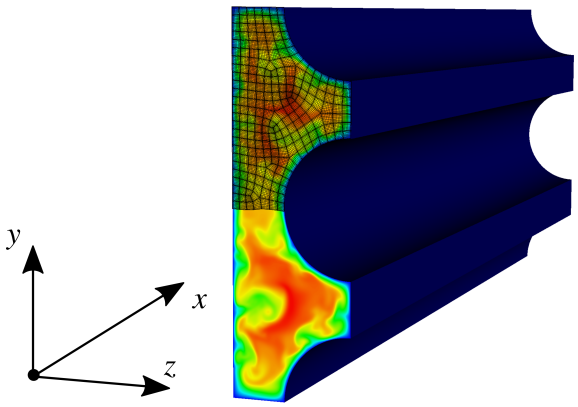

Figure 1: Scheme of DNS calculation geometry and computational grid.

DNS calculations of the passive scalar field c (temperature) were performed in the peripheral fragment of the rod bundle channel [18] with Re = 4270 chosen and at seven different Prandtl numbers Pr = 0.025, 0.05, 0.1, 0.2, 0.4, 0.6, 0.71 to provide

training data for the modified model of the turbulent scalar flux (TSF) vector using ML methods. At the left boundary (see Fig. 1), the smaller value of the scalar $c = 0$ is set, while the right boundary, which includes fragments of three rods, has the larger value $c = 1$. As for the velocity field, periodic boundary conditions are applied to the scalar c along the flow direction, which produces a two-dimensional field $\overline{c}(y, z)$ after averaging over t and x. Preliminary results of the simulations of the mean scalar field $\overline{c}$ and those of determining the components of the turbulent scalar flux vector $\overline{u_i' c'}$ using DNS in the peripheral fragment of the channel, as well as various RANS approximations for $\overline{u_i' c'}$ are presented in [18].

3 Methods to develop modified RANS models for heat problems using ML

In modern turbulence models, the turbulent scalar flow approximation to close the equation for the mean scalar $\overline{c}$ can be formulated as follows [17]:

$$-\overline{u_i' c'} = D_{ij}\frac{\partial \overline{c}}{\partial x_j} = D^*_{ij}\nu_t\frac{\partial \overline{c}}{\partial x_j} = (\frac{\nu_t}{Pr_t}\delta_{ij} + D^+_{ij}\nu_t)\frac{\partial \overline{c}}{\partial x_j} \tag{1}$$

$$D_{ij} = \sum_{n=1}^{6} g^{(n)}(I_1, I_2, ..., I_{15})T^{(n)}_{ij}. \tag{2}$$

Here, the factors D_{ij} and D^*_{ij} or their correction with the factor D^+_{ij} to the gradient turbulent diffusion hypothesis (GDH), where $Pr_t = 0.85$, represent a parametric term in the form of a linear combination of basis tensors, which is optimized using DNS data. The tensor basis $T^{(n)}_{ij}$ is obtained explicitly from the dimensionless deformation and rotation tensors s_{ij} and r_{ij} [6, 17]:

$$T^{(1)}_{ij} = \delta_{ij},\ \ T^{(2)}_{ij} = s_{ij},\ \ T^{(3)}_{ij} = r_{ij}, \tag{3}$$

$$T^{(4)}_{ij} = s_{ik}s_{kj},\ \ T^{(5)}_{ij} = r_{ik}r_{kj},\ \ T^{(6)}_{ij} = s_{ik}r_{kj} + r_{ik}s_{kj},$$

The expressions for the scalar invariants I_m $(m = 1, ..., 15)$ taken from the previous study [17] also contain the mean scalar gradient and other quantities:

$$I_1 = tr(s_{ik}s_{kj}),\ \ I_2 = tr(s_{lm}s_{mn}s_{nk}),\ \ I_3 = tr(r_{ik}r_{kj}),\ \ I_4 = tr(s_{lm}r_{mn}r_{nk}),$$

$$I_5 = tr(s_{lm}s_{mn}r_{nk}r_{kp}),\ \ I_6 = tr(s_{lm}s_{mn}r_{nk}r_{kp}s_{pq}s_{ql}),\ \ I_7 = \nabla\overline{c}^T\nabla\overline{c},\ \ I_8 = \frac{\partial \overline{c}}{\partial x_j}^T s_{ij}\frac{\partial \overline{c}}{\partial x_i},$$

$$I_9 = \left(\frac{\partial \overline{c}}{\partial x_j}\right)^T s_{ik}s_{kj}\frac{\partial \overline{c}}{\partial x_i},\ \ I_{10} = \left(\frac{\partial \overline{c}}{\partial x_j}\right)^T r_{ik}r_{kj}\frac{\partial \overline{c}}{\partial x_i},\ \ I_{11} = \left(\frac{\partial \overline{c}}{\partial x_j}\right)^T s_{ik}r_{kj}\frac{\partial \overline{c}}{\partial x_i}, \tag{4}$$

$$I_{12} = \left(\frac{\partial \overline{c}}{\partial x_j}\right)^T s_{ik}s_{kl}r_{lm}\frac{\partial \overline{c}}{\partial x_i}, I_{13} = \left(\frac{\partial \overline{c}}{\partial x_j}\right)^T r_{ik}s_{kl}r_{lm}r_{mj}\frac{\partial \overline{c}}{\partial x_i},$$

$$I_{14} = Re_d,\ \ I_{15} = \nu_t/\nu,$$

where $Re_d = \sqrt{k}W_d/\nu$ corresponds to the local Reynolds number based on the distance to the nearest wall W_d, and k is the turbulent kinetic energy.

Using the MGEP evolutionary algorithm as a basis, it is possible to build explicit algebraic expressions for approximating the turbulent scalar flux based on high-fidelity DNS data. Two of the simple expressions found with the minimal TSF estimation error have contained the I_{14} feature in the expression for $D^*_{ij} = D^*_{ij}(I_{14})$ [18]:

$$-\overline{u'_i c'} = (0.27 + 0.40 I_{14}) \frac{\nu_t}{Pr_t} \frac{\partial \overline{c}}{\partial x_i}. \tag{5}$$

$$-\overline{u'_i c'} = (0.27 + 0.40 I_{14} + 0.25 I_1 I_{14}) \frac{\nu_t}{Pr_t} \frac{\partial \overline{c}}{\partial x_i}$$

Another approach for estimating the TSF vector, using the UIML-s neural network model, has been proposed by the authors in the same paper [18], where the results of prediction of the mean scalar field using the classical GDH approximation, the model obtained using the MGEP symbolic regression, two neural network models, TBNN-s from [17] and its modification using UIML-s have been compared. The main differences between the UIML-s model and TBNN-s were the implicit generation of a tensor basis from the tensors s_{ij} and r_{ij} during model training and the introduction of residual connections for better passage of the gradient into the internal layers of the model during training.

When training closure models, all input data with invariants and basis tensors are taken from the corresponding RANS calculations [18] in the upper half of the domain illustrated in Fig. 1, and the high-fidelity output data that the model should approximate are taken from the concurrent DNS calculations [15, 18].

4 Methods to interpret obtained machine learning models

In this paper, two methods are used to analyze the importance of invariants fed to a turbulent scalar (heat or mass) flux model. Both methods are independent of the type of machine learning models, i.e. they can be used equally with models of different types, be it an ensemble of decision trees, a neural network structure, or, for example, a model based on symbolic expressions. However, the first permutation method to assess the importance of PFI (Permutation Feature Importance [19]) is global and estimates the behavior of the model on average over the entire data set. The second method, coming from the theory of cooperative games in economics, based on Shapley values and called SHAP (SHapley Additive exPlanations [13]), has the ability to explain individual predictions made by the model.

The basic idea of the permutation method is simple and consists of estimating the increase in the prediction error of the model f on the permutation of the order of observations of invariants in the feature matrix X_{perm} relative to the original error on the original input X:

$$FI = L^{-1}(y, f(X)) - L^{-1}(y, f(X_{perm})), \tag{6}$$

where $L(y, f)$ is the error measure. The more important the input invariant, the higher the FI value due to the increase in the error on the feature matrix with permutation

of observations for this invariant. It is argued that if the presence of an invariant in the input features does not matter, then the model error will not change when its values are permuted and the FI value will be zero. In practice, the resulting values FI are normalized over the sum of the importance values for all input features for a convenient presentation of these values in fractions $\hat{FI}_i = FI_i / \sum FI_i$. An inconvenience in the permutation method is the need, in addition to the type of model analyzed f, to know the exact values of the model y in the input data, so the analysis is usually carried out on the training data set.

A more flexible method to interpret machine learning models is the SHAP method, since it allows interpreting the work of the model on each specific example of data $x^{(0)}$, consisting of the current set of features M. It is based on assessing the contribution of each i-th feature (here, of the Shapley value) ϕ_i into the model value, so that additivity is satisfied:

$$f(x^{(0)}) = \phi_0 + \sum_{i=1}^{M} \phi_i x'_i, \tag{7}$$

where x' is a simplified (binary) set of features of the original model $f(x)$ with a mapping that transforms the simplified vector of zeros and ones into the original feature vector $x = h_x(x')$, so that $x^{(0)} = h_x([1, 1, ..., 1])$, $E[x] = h_x([0, 0, ..., 0])$ and $\phi_0 = f(E[x])$ is the base value of the model in the absence of all features. If the corresponding $x'_i = 0$, it means that this feature is absent from the model, that is, $\phi_i = 0$. In such a case, it can either be physically absent with retraining of the model in the data set without it, or replaced with a constant value of feature i equal to the mathematical expectation in the data set considered $E[x_i]$. We can introduce the notations $f_x(z') = f(h_x(z'))$ and $z' \setminus i$ (which means setting $z'_i = 0$ in the simplified vector z'), then the classical expression for the contribution of each feature in the SHAP method will be as follows:

$$\phi_i(f, x) = \sum_{z' \subseteq x'} \frac{|z'|!(M - |z'| - 1)!}{M!} [f_x(z') - f_x(z' \setminus i)], \tag{8}$$

where $|z'|$ is the number of non-zero components in the vector z'.

In practice, disabling the i-th feature in the original model means retraining the model on all subsets of the feature sets without i. This is time-consuming, so the values of ϕ_i are usually estimated using simplified expressions. It is possible to calculate the approximate Shapley value in the expression much faster if you do not count all the elements of the sum (8), but only some of them, which have large weights. For this, the weights are used as probabilities when sampling the elements of the sum (approximation by sampling). However, even in this case, it is necessary to calculate the conditional mathematical expectations, which, in the case of statistical independence of the features and linearity of the model in the neighborhood of $x^{(0)}$, can be approximated by the value averaged over the data sample $E[f_x(z'|z'_i = 0)] \approx f_x(E[z'|z'_i = 0])$. In the Kernel SHAP estimation method used in this paper, the local estimation of the linear model g explaining the target model f is performed by the least squares method similar to the LIME method [12], where the original analyzed model f is locally approximated

by an unknown linear model g near $x^{(0)}$ using regression:

$$\sum_{i=1}^{N} w^{(i)}[f(x^{(i)}) - g(x^{(i)})]^2 + \Omega(g) \to \min_g, \tag{9}$$

where the weights of examples $w^{(i)}$ from the training set of length N as a measure of their closeness to $x^{(0)}$ are chosen empirically, and $\Omega(g)$ is a penalty for the complexity of the approximating function, for example, the number of non-zero weights in a linear model. Once the approximating linear model g is obtained, analyzing the importance of the features is easy. When switching from LIME to the Kernel SHAP method, the corresponding weights (kernel) in the least squares method are written as follows [13]:

$$\sum_{z' \in Z} \frac{(M-1)}{C_M^{|z'|}|z'|(M-|z'|)}[f_x(z') - g(z')]^2 \to \min_g . \tag{10}$$

The Kernel SHAP method can be applied to any type of ML model, but there are variations with simplified calculations for various specific types of models, such as Deep SHAP neural networks, Tree SHAP ensemble-based models. The limitation of designated SHAP methods for reliable feature importance assessment is their statistical independence, which always requires additional verification.

5 Results

The two methods of model interpretation described in the previous section are used to assess the importance of input invariants in the approximations of the turbulent scalar (heat) flux components formulated in [18]. Fig. 2 shows the diagrams of the importance of input features in the MGEP expression for the TSF vector obtained by the SHAP method. It is evident that the most important feature with a non-zero weight value for this approach is I_{14}. These results are in complete agreement with simple and human-readable expressions (5) obtained using the MGEP algorithm, where the invariant I_{14} is included as an input feature. This trivial example is given here to demonstrate the adequacy of the results obtained by the SHAP method.

Fig. 3 shows the feature importance diagrams for neural network models for two transverse components of the TSF vector, $\overline{v'c'}$ and $\overline{w'c'}$, obtained by the SHAP method. The most important features are seen to be I_{14}, I_{15}, I_2, I_4 for TBNN-s and I_{15}, I_5 for UIML-s, if we restrict ourselves to the average Shapley value as half of the maximum. Invariants that contain scalar gradient values have a reduced significance in models, and the least significant are $I_{10} = \left(\frac{\partial \overline{c}}{\partial x_j}\right)^T r_{ik} r_{kj} \frac{\partial \overline{c}}{\partial x_i}$ for TBNN-s and $I_7 = (\nabla \overline{c})^T \nabla \overline{c}$ for UIML-s.

If we look at the feature importance assessment using the permutation PFI method, the results of which are shown in Figs. 4 and 5, we see a similar pattern with the most important invariants I_{14} and I_{15} among the dominant ones. We note a distinctive feature of the PFI method: the presence of negative values of relative importance $\hat{FI}_i$ at the bottom of Figs. 4 and 5 for the component $\overline{v'c'}$, which are formed due to the decrease in the absolute error of the model when permuting observations, for example,

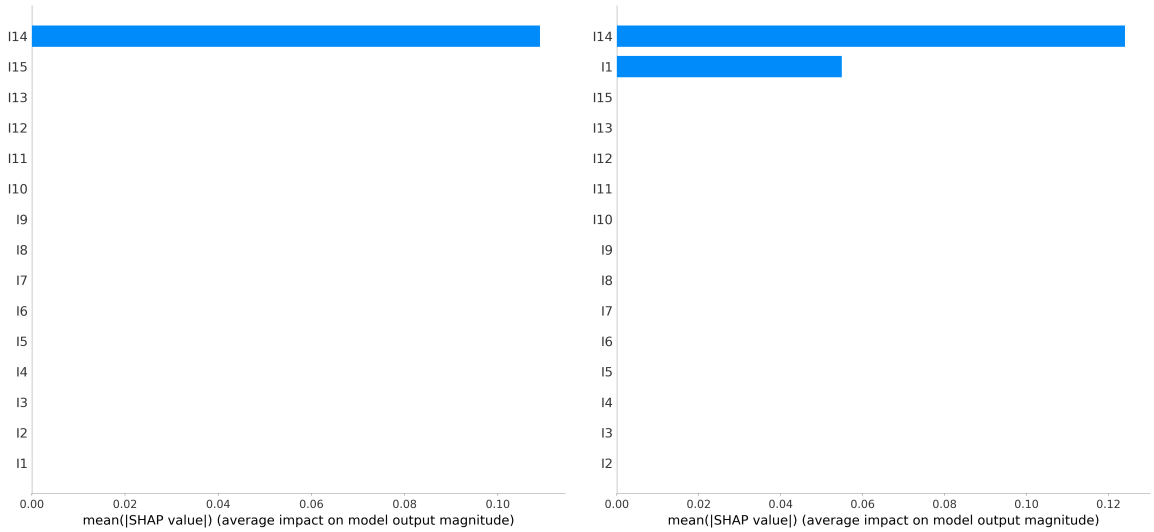

Figure 2: Feature importance in terms of mean SHAP values for the MGEP models (5) over all features for all TSF components.

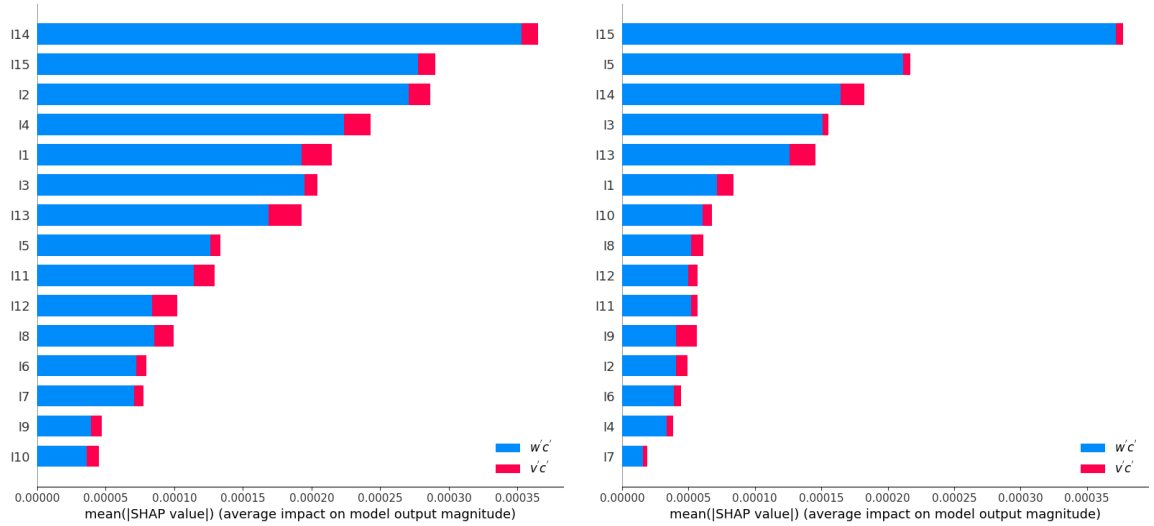

Figure 3: Feature importance in terms of mean SHAP values for TBNN-s (left) and UIML-s (right) models over all features for two cross-stream TSF components.

for invariants I_{10}, I_{11}, I_3, I_5, and I_1. This effect will be studied in future papers. Apart from this, the results of the simple permutation analysis method are consistent with the results of the SHAP method.

The influence of the correlation of the input invariants on the result of the analysis of the importance of specific features in the model is then evaluated. For this purpose, the matrix of Pearson correlation coefficients of features with each other is calculated (see Fig. 6, on the right), where invariants are grouped using the Ward connection into clusters by mutual correlation with each other, which can be observed in the form of square blocks of increased correlation with values close to 1 and -1 along the

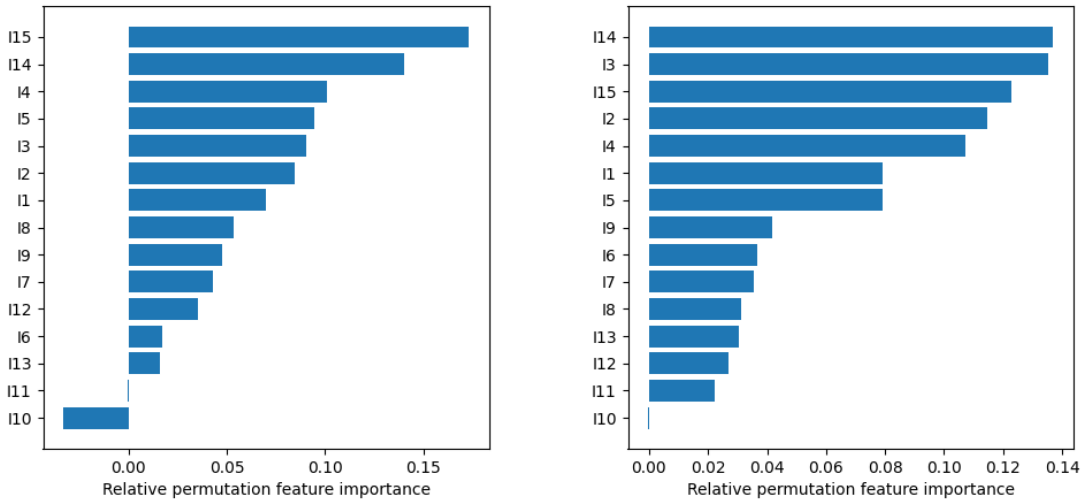

Figure 4: Feature importance in terms of relative PFI values for TBNN-s model and TSF components, $\overline{v'c'}$ (left) and $\overline{w'c'}$ (right).

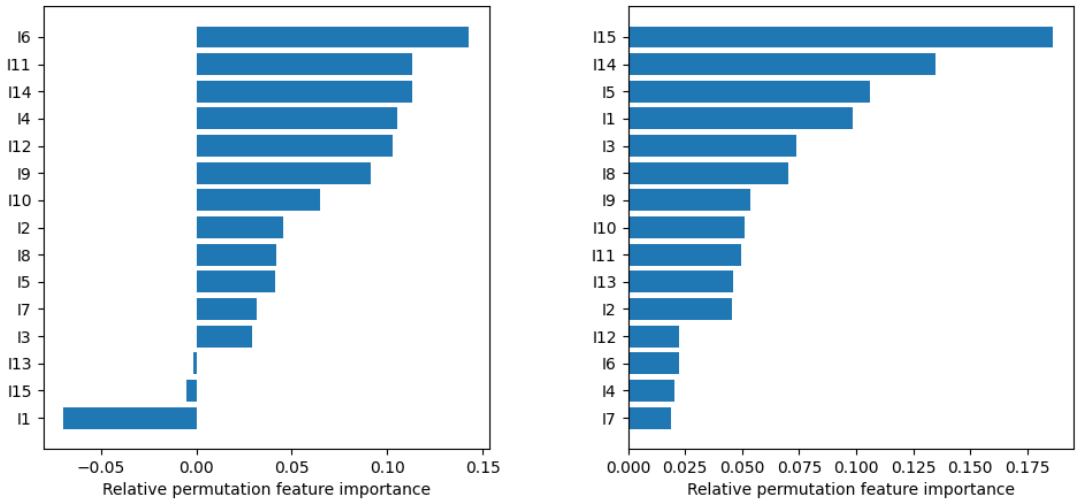

Figure 5: Feature importance in terms of relative PFI values for UIML-s model and TSF components, $\overline{v'c'}$ (left) and $\overline{w'c'}$ (right).

diagonal of the matrix. Fig. 6, on the left, shows a dendrogram showing the hierarchy of correlation coefficients for features for the considered matrix. Using this hierarchy, five clusters with features dependent on each other within a cluster at the 0.5 level for the dendrogram are selected:

1. $(I_{13}, I_9, I_{10}, I_{11})$ with a typical representative $I_{13} = \left(\frac{\partial \overline{c}}{\partial x_j}\right)^T r_{ik} s_{kl} r_{lm} r_{mj} \frac{\partial \overline{c}}{\partial x_i}$;
2. (I_{14}, I_7, I_{15}) with a representative $I_{14} = Re_d$;

3. (I_5, I_1, I_3) with a representative $I_1 = tr(s_{ik}s_{kj})$;

4. (I_8, I_{12}) with a representative $I_8 = \left(\frac{\partial \bar{c}}{\partial x_j}\right)^T r_{ik}s_{ij}\frac{\partial \bar{c}}{\partial x_i}$;

5. (I_6, I_2, I_4) with a representative $I_2 = tr(s_{lm}s_{mn}s_{nk})$.

Invariants representing the corresponding clusters have been selected based on the highest importance values from the previous analysis in Fig. 3. The SHAP analysis was repeated for the neural network models re-trained with a limited set of features with the specified representatives from each cluster. For the first TBNN-s model (see Fig. 7), the invariant with the highest importance value changed from I_{14} to I_1, so cluster 2 of correlated features after removing statistically dependent values gave preference to cluster 3 including invariants I_5, I_1, I_3 and moved down to the second position.

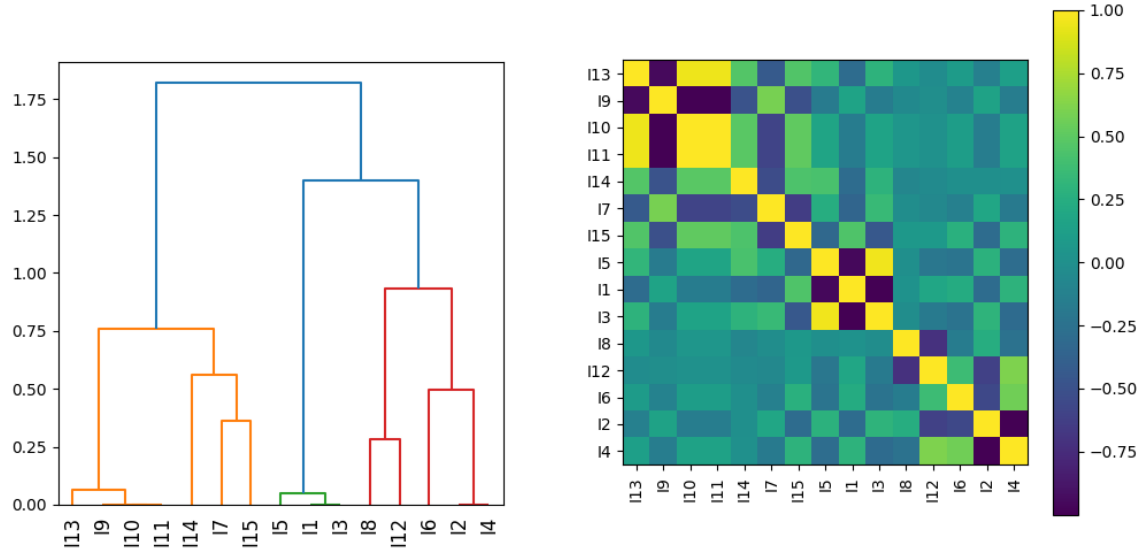

Figure 6: Dendrogram for the correlation distance between different features clustered using the Ward's linkage (left) and the correlation heatmap (right).

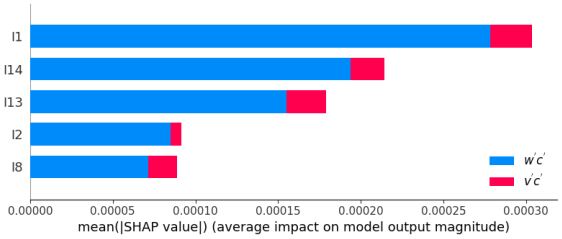

Figure 7: Feature importance in terms of mean SHAP values for TBNN-s model over five selected features according to the clustering for TSF components, $\overline{v'c'}$ and $\overline{w'c'}$.

However, if we look at the analysis of the simplified UIML-s model with a reduced number of dependent features (see Fig. 8), the picture does not change much, and

cluster 2 remains decisive for the output values of the $\overline{w'c'}$ component of the TSF vector. Here we can conclude that the correlation of features for the new UIML-s model has little effect on the result of assessing their significance in a complex model using the SHAP method based on the Kernel SHAP algorithm. The application of the permutation PFI method for simplified neural network models to assess the turbulent scalar flux with a reduced number of invariants showed similar results to those of the SHAP method, so they are not presented here.

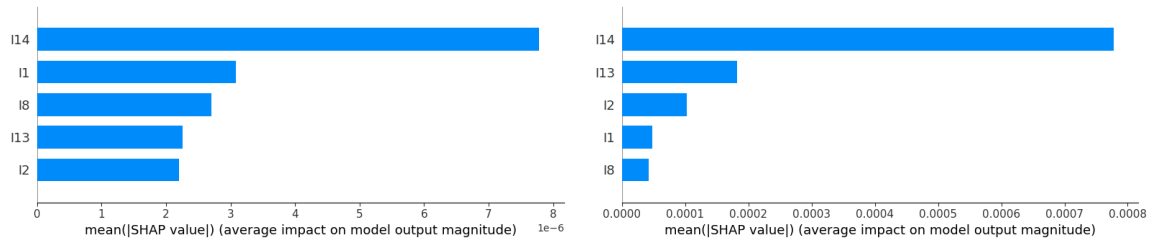

Figure 8: Feature importance in terms of mean SHAP values for UIML-s model over five selected features according to the clustering for TSF components, $\overline{v'c'}$ (left) and $\overline{w'c'}$ (right).

In addition, the importance analysis of the input basis tensors in the TBNN-s model was performed using the PFI method, which allows us to group any subset of the input values of the analyzed model and permute the examples for the group as for a single object, for example, to group the values of the tensor components. The result of such an analysis is shown in Fig. 9, where the relative values of $\hat{FI}$ among the tensors are shown for the TBNN-s model, which is based on six basis tensors. Here, as can be seen, the tensor $T_4 = s_{ik}s_{kj}$ is decisive in calculating the values of the component $\overline{w'c'}$. The zero values of the contribution of the tensor $T_1 = I$, identically equal to the identity matrix, can be explained by the peculiarity of the PFI method. The point is that in this case, when permuting constants in data examples, the matrix of input features does not change its value, and the inverse of the error in the matrix of features X_{perm} in the formula (6) will be identically equal to the inverse of the error on the original matrix of features X. Therefore, other methods must be used to analyze the influence of constant features. The permutation of examples in a group associated with the values of the tensor components $T_{ij}^{(3)} = r_{ij}$ again leads to an improvement in the accuracy of the model, as in the case described above with some invariants. At the moment, the question remains why optimization of the parameters of the TBNN-s model by the stochastic gradient descent algorithm, for example, does not automatically reduce the contribution of the tensor $T_{ij}^{(3)}$ during training. Perhaps this is due to the balance of the loss function of the model by the second component $\overline{v'c'}$. The confirmed basis tensor on which the turbulent scalar flux depends weakly is $T_{ij}^{(6)}$.

The advantage of the UIML-s model over TBNN-s is that it is fed only two initial tensors s_{ij} and r_{ij} at the input, and the basis tensors are automatically output during model training. Therefore, the analysis of the importance of the input tensors for the UIML-s model does not make much sense, although, for testing purposes, it was also carried out and the contribution of the tensor $r_{ij} = 0.24$ and $s_{ij} = 0.76$, which is

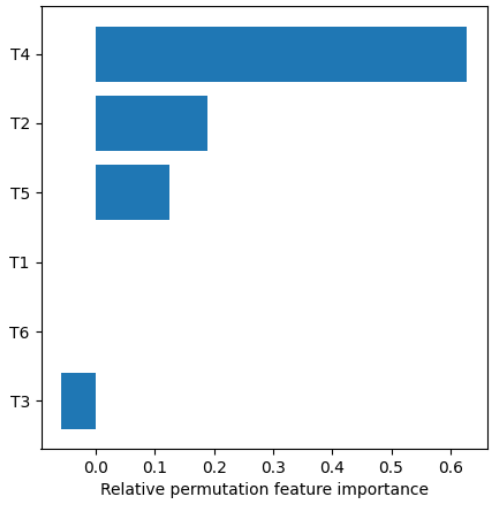

Figure 9: Importance of TBNN-s tensor basis elements for predictions of the TSF component $\overline{w'c'}$ in terms of relative PFI values.

somewhat at odds with the result for the TBNN-s model presented above.

Thus, the development of new RANS-ML models for heat and mass transfer problems can be organized as follows: prepare a data set for a given flow geometry with a redundant set of invariants, train the UIML-s model on this set of features, analyze the importance of the contribution of each feature using the PFI or SHAP methods on a complex model. At the final stage, obtain a simplified expression of the model using the MGEP method, convenient for modifying the RANS solver of the CFD code on a limited set of invariants obtained from the result of interpreting the contribution of each feature.

6 Conclusion

In this paper, we perform a feature importance analysis to improve the understanding of turbulent scalar (heat or mass) flux approximations obtained using MGEP, TBNN-s, and UIML-s machine learning methods for the scalar field in a turbulent channel flow with complex geometry simulating the peripheral fragment of a rod bundle channel. Two methods for interpreting and explaining the machine learning models obtained are considered. The first method, PFI, estimates the model behavior on average throughout the data set. The second method, SHAP, allows us to explain the individual predictions made by the model. The importance of using the Re_d invariant for predicting the TSF vector in the geometry considered is shown. The results are confirmed by the SHAP and PFI methods for all machine learning models. The effect of the correlation of input invariants on the result of the feature importance analysis by the SHAP and PFI methods in neural network models to estimate the TSF vector is assessed. Based on the conducted studies, a method to organize the process of developing interpretable turbulence models for heat and mass transfer problems using machine learning methods

is proposed.

Acknowledgment

The work was carried out with the financial support of the Russian Science Foundation, project no. 22-19-00587, `https://rscf.ru/en/project/22-19-00587/`, the computational resources for DNS simulations were provided under a state contract with IT SB RAS.

References

[1] Spalart P.R., *Strategies for turbulence modelling and simulations*, Int. J. Heat Fluid Flow, Vol. 21, 2000, P. 252–263.
[2] Durbin P., *Some recent developments in turbulence closure modeling*, Annu. Rev. Fluid Mech, Vol. 50, 2018, P. 77–103.
[3] Duraisamy K., Iaccarino G., Xiao H., *Turbulence modeling in the age of data*, Annu. Rev. Fluid Mech, Vol. 51, 2019, P. 357–377.
[4] Vinuesa R., Brunton S.L., *Enhancing computational fluid dynamics with machine learning*, Nature Computational Science, Vol. 2, 2022, P. 358–366.
[5] Weatheritt J., Sandberg R. D. *A novel evolutionary algorithm applied to algebraic modifica-tions of the RANS stressвЂ“strain relationship* J. Comput. Phys., Vol. 325, 2016, P. 22–37.
[6] Ling J., Kurzawski A., Templeton J., *Reynolds averaged turbulence modelling using deep neural networks with embedded invariance*, J. Fluid Mech., Vol. 807, 2016, P. 155вЂ“166.
[7] Jiang C., Vinuesa R., Chen R., Mi J., Laima S., Li H., *An interpretable framework of data-driven turbulence modeling using deep neural networks*, Phys. Fluids, Vol. 33, 2021, 055133.
[8] McConkey R., Yee E., Lien F.S., *Deep structured neural networks for turbulence closure modeling*, Phys. Fluids, Vol. 34, 2022.
[9] Yakovenko S.N., Razizadeh O., *Machine learning methods for development of data-driven turbulence models*, AIP Conf. Proc., 2288, 2020, 030065.
[10] Pope S.B., *A more general effective-viscosity hypothesis*, J. Fluid Mech., Vol. 72, 1975, P. 331–340.
[11] He X., Tan J., Rigas G., Vahdati M., *On the explainability of machine-learning-assisted turbulence modeling for transonic flows*, Int. J. Heat Fluid Flow, Vol. 97, 2020, P. 1–16.
[12] Ribeiro M.T., Singh S., Guestrin C., *Why should I trust you?: Explaining the predictions of any classifier*, Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. ACM, 2016, P. 1135–1144.
[13] Lundberg S.M., Lee S.I., *A unified approach to interpreting model predictions*, Advances in neural information processing systems, Vol. 30, 2017, P. 1–10.
[14] Mandler H., Weigand B., *Feature importance in neural networks as a means of interpretation for data-driven turbulence models*, Computers & Fluids, Vol. 265, 2023.
[15] Ivashchenko V.A., Lobanov P.D., Yavorsky N. I., Tokarev M.P., Mullyadzhanov R.I., *Direct numerical simulation of the peripheral and internal configurations of a model assembly of fuel elements*, J. Appl. Ind. Math., Vol. 17, 2023, P. 320вЂ“325.
[16] Fischer P.F., Lottes J.W., Kerkemeier S.G., *Web page, http://nek5000.mcs.anl.gov*, 2008.
[17] Milani P.M., Ling J., Eaton J.K., *Turbulent scalar flux in inclined jets in crossflow: counter gradient transport and deep learning modelling*, J. Fluid Mech., Vol. 906, 2021, A27.
[18] Li H., Yakovenko S., Ivashchenko V., Lukyanov A., Mullyadzhanov R., Tokarev M., *Data-driven turbulence modeling for fluid flow and heat transfer in peripheral subchannels of a rod bundle*, Phys. Fluids, Vol. 36, 2024, 025141.
[19] Breiman L. *Random Forests* Machine Learning, Vol. 45, 2001, P. 5–32.

Tokarev M.P.,
Institute of Thermophysics SB RAS,
Russia, 630090, Novosibirsk, Academician Lavrentiev Avenue, 1,
Email: mtokarev@itp.nsc.ru

Yakovenko S.N.,
Institute of Thermophysics SB RAS,
Russia, 630090, Novosibirsk, Academician Lavrentiev Avenue, 1,
Email: s.yakovenko@mail.ru

Lukyanov A.A.,
Institute of Thermophysics SB RAS,
Russia, 630090, Novosibirsk, Academician Lavrentiev Avenue, 1,
Email: a.lukyanov@alumni.nsu.ru

Received 06.09.2024, Accepted 08.11.2024, Available online 31.12.2025.