Applications in Energy and Combustion Science 11 (2022) 100077

Contents lists available at ScienceDirect

Applications in Energy and Combustion Science

journal homepage: www.elsevier.com/locate/jaecs

Gradient boosted decision trees for combustion chemistry integration

S. Yao [a,c,d,*], A. Kronenburg [b], A. Shamooni [b], O.T. Stein [b], W. Zhang [a,c,d]

[a] Ningbo Institute of Materials Technology and Engineering, Chinese Academy of Sciences, 315201 Ningbo, China
[b] Institut für Technische Verbrennung, Universität Stuttgart, Pfaffenwaldring 31, 70569 Stuttgart, Germany
[c] Key Lab. of Aero Engine Extreme Manufacturing Technology of Zhejiang Province, 315201 Ningbo, China
[d] University of Chinese Academy of Sciences, 100049 Beijing, China

ARTICLE INFO

Keywords:
Ensemble learning
Gradient boosting
Chemical kinetics
Hydrogen combustion

ABSTRACT

This study introduces the gradient boosted decision tree (GBDT) as a machine learning approach to circumvent the need for a direct integration of the typically stiff system of ordinary differential equations that govern the temporal evolution of chemically reacting species. Stiffness primarily relates to the chemistry integration and here, hydrogen/air systems are taken to train and test the ensemble learning approach. We use the LightGBM (Light Gradient Boosting Machine) algorithm to train GBDTs on the time series of various self-igniting mixtures from the time of ignition to equilibrium composition. The GBDT model provides reasonable predictions of the species compositions and thermodynamic states at the next time step in an a priori study. A much more challenging a posteriori study shows that the model can reproduce a full time–history profile of the igniting H_2/air mixtures, as the results agree very well with those obtained from a direct integration of the ODEs. The GBDT model can be deployed as standalone C++ codes and a speed-up by one order of magnitude has been demonstrated. The GBDT approach can thus be considered as an efficient method to represent the chemical kinetics in the simulation of reactive flows. It provides an alternative to deep artificial neural networks (ANNs) that is comparable in accuracy but easier to couple with existing CFD codes.

1. Introduction

The ordinary differential equations (ODEs) that describe the temporal evolution of the composition space due to chemical reactions tend to be stiff and their integration can be computationally expensive. This issue is exacerbated by the constantly increasing size of the state-of-the-art chemical mechanisms that can contain thousands of reacting species with ten thousands of chemical reactions. Strategies to reduce the computational burden include chemistry reduction, tabulation and – more recently – machine learning approaches. Especially the latter has been of growing interest and an increasing number of studies has been directed towards the exploitation of deep artificial neural networks (ANNs) to reduce the cost of solving the time evolution of the species' compositions. The attempt to use ANNs for chemical kinetics started with the pioneering work of Blasco et al. [1] where shallow ANNs were used to reproduce the temporal evolution of a reduced hydrocarbon mechanism. With the rapid progress of deep learning techniques and the advent of powerful open-sourced frameworks like TensorFlow [2], this methodology has been extended to numerous applications in combustion modeling where ANNs have been widely used to address the necessity of directly solving the systems of ODEs. For example, the ANN method has been used for chemistry tabulation or direct integration of the chemical source terms where the numerical solvers were replaced by ANN-based solutions to compute the species compositions and corresponding reaction rates [3–10]. In the latest studies [11–14] various architectures of ANNs, such as the convolutional neural networks (CNNs) [15] and the residual neural networks (ResNets) [16], have been introduced for a potential reduction of predictive errors. Also, there is a trend to combine the principal component analysis (PCA) with ANNs [17,18], where a high-dimensional space consisting of the species compositions and thermodynamic states is mapped to a low-dimensional manifold for the reduction of dimensions of the input layer in ANNs. Also, clustering techniques such as the self-organizing map (SOM) [19] were introduced in Refs. [4–6] to partition the composition space into sub-zones and each sub-zone is then associated with an individual ANN to ensure accuracy.

Although not being as popular as deep learning approach for combustion modeling, ensemble learning and decision tree based models are considered to be as powerful and flexible as deep learning and witnessed similar rapid advances in recent years as ANNs. Chung et al. [20], for example, used decision trees to find the optimal combustion models at different locations of the computation domain. Instead of using the finite-rate chemistry or the flamelet method [21] over

* Corresponding author at: Ningbo Institute of Materials Technology and Engineering, Chinese Academy of Sciences, 315201 Ningbo, China.
E-mail address: yaosongbai@nimte.ac.cn (S. Yao).

https://doi.org/10.1016/j.jaecs.2022.100077
Received 2 March 2022; Received in revised form 10 June 2022; Accepted 25 July 2022
Available online 3 August 2022

the entire domain, the local chemical kinetics were represented by a corresponding combustion model determined by decision trees to achieve a trade-off between accuracy and efficiency. Ensemble learning and decision trees have also been used in the presumed probability density function (PDF) method for LES of reacting flows [22]. In our previous study [23], the state-of-the-art ensemble learning algorithms were introduced for the closure of the PDF and conditional scalar dissipation rate of mixture fraction in turbulent sprays, and found to be a promising technique for combustion related modeling. As a machine learning technique that differs considerably from deep ANNs, ensemble learning poses certain properties that make it advantageous. First of all, ensemble learning is highly optimized for parallel computing using central processing units (CPUs), whereas ANNs rely heavily on graphics processing units (GPUs). Considering that the majority of computational fluid dynamics (CFD) codes run on CPUs and that the power of ANNs is fully unleashed on GPUs, the hardware requirements for using ensemble learning in CFD codes are much more favorable. Besides, the ensemble learning algorithm is found to be more robust to outliers, less insensitive to transformations of inputs, and higher computational scalability for large datasets [24]. In light of the above, ensemble learning shall be considered for expanding the scopes of machine learning applications in combustion modeling, and our study is the first attempt to demonstrate the possibility of representing the chemical kinetics efficiently using ensemble learning and its implementation in a standard CFD software package.

2. Tree models and ensemble learning

The concept of ensemble learning is that a strong and high predictive model can be generated by combining multiple base learners, i.e., a decision tree is built. Note that the decision tree is significantly different from the Euclidean minimum spanning tree (EMST) [25], which is used to determine the neighboring particles of an interacting particle. In contrast, the decision tree works as a general regression function and is similar to the ANN model for an arbitrary function $f(\boldsymbol{x};\boldsymbol{\omega},\boldsymbol{b}) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{\omega}+\boldsymbol{b}$, where $\boldsymbol{\omega}$ and $\boldsymbol{b}$ are trainable parameters. Similar to the ANN, a decision tree is rule-based but the output is computed by conditional statements and not by mathematical functions. Fig. 1 provides an example where – for the sake of simplicity – we assume the temperature to be predictable by the concentration of species. The decision tree starts from a single root node and grows by splitting the training samples (input–output pairs) based on the input variables. In real applications, the decision tree will continue to grow in depth.

For the present study, the input variables will be the species composition (in mass fractions), pressure and temperature at time t, and the output will be the new composition state to which the chemical system moves forward during the time step Δt. When the tree grows, internal nodes or *branches* are created to hold subsets of the training samples. The decision tree will continue to grow in search for the best splitting policy until every training sample is assigned to a terminal node or *leaf* where a prediction is made by averaging the output of the subset training samples on the terminal leaf. It should be pointed out that every time the tree splits, all input variables are taken into consideration. From them, one is selected that provides the best splitting strategy. In other words, an input variable can appear multiple times in the conditional statements. Eventually, the decision tree is trained to learn an auto-regressive function of $[T, p, Y_i]_t \rightarrow [T, p, Y_i]_{t+\Delta t}$. For ANNs there should not be just one but multiple layers and numerous neurons in each layer in order to build a complex model that satisfies the concepts of deep learning. By the same token, it is hard to make accurate predictions if there is only one decision tree. We therefore need to grow a number of trees and there are a family of algorithms to determine how those trees are gown and organized, typified by the bootstrap aggregating (or Bagging for short) and gradient boosting techniques.

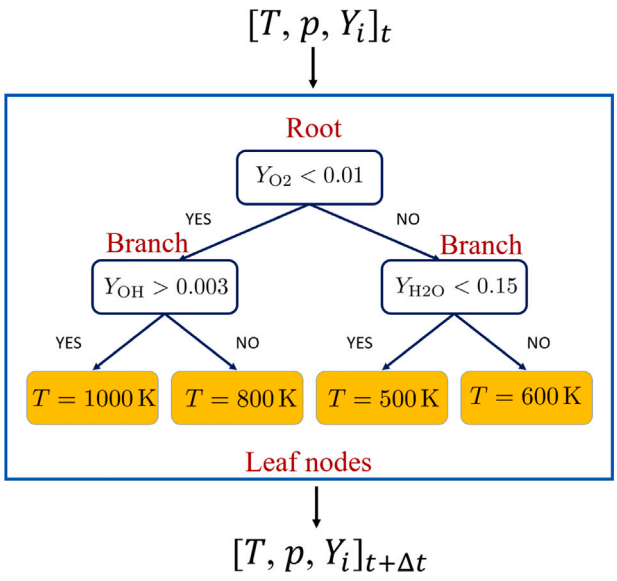


Fig. 1. Schematic of decision trees and tree-based ensemble learning.

The conventional ensemble learning framework, Random Forests [26], uses the bootstrap aggregating technique. It draws a random subset from the training dataset (sampling with replacement) for the training of each decision tree. Each decision tree is trained independently. The ensemble model then makes predictions by simply aggregating the predictions of all base learners (see Fig. 2), i.e., it takes the average of all (independent) predictions or uses a majority voting rule. In contrast, the boosting algorithms, i.e. XGBoost (eXtreme Gradient Boosting) [27] and LightGBM (Light Gradient Boosting Machine) [28], train decision trees in a sequential manner, meaning that each tree grows after the other sequentially, as shown in Fig. 2. Specifically, a subsequent learner will learn to minimize the residual errors (the errors between the prediction and the true value) made by its predecessor. Such types of decision trees are called gradient boosted decision trees (GBDTs).

Similar to the training algorithm of ANNs, which updates the trainable parameters to minimize a *loss* function, the boosting algorithm [27] grows the decision trees by minimizing the following objective

$$\mathcal{L}(\phi) = \sum_i l\left(\hat{y}_i, y_i\right), \tag{1}$$

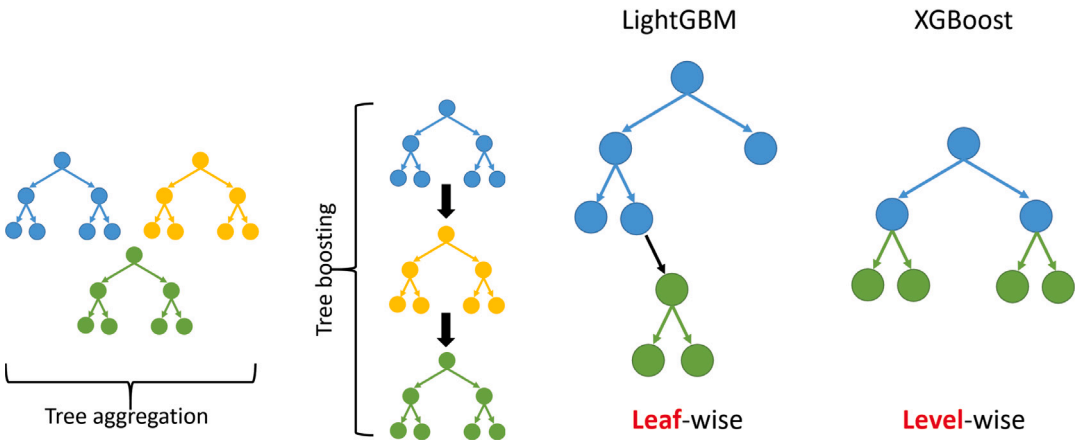
where l is the loss function that measures the mean squared error between the prediction $\hat{y}$ and the true value y. GBDTs use a greedy algorithm to search for split candidates that is equivalent to the reduction of $\mathcal{L}(\phi)$.

LightGBM and XGBoost follow the same principle except that LightGBM uses a leaf-wise policy to grow GBDTs, which splits the tree only on the best nodes that can bring maximum reduction of the loss function, whereas XGBoost implements a level-wise policy that leads to a symmetrical tree structure (as illustrated in Fig. 2(b)). GBDTs grow much faster using the leaf-wise policy, and with the extra help of techniques offered by LightGBM, i.e., gradient-based one-sided sampling (GOSS) and exclusive feature bundling (EFB) [28], the speed of training is highly accelerated. We therefore select LightGBM to train the GBDT model.

3. Results and discussion

3.1. Datasets and GBDT model

A standard hydrogen mechanism which consists of 9 species (H_2, O_2, O, OH, H_2O, H, HO_2, H_2O_2, and N_2) and 19 reactions [29] is



(a) Bootstrap aggregating and gradient boosting techniques

(b) Types of GBDTs

Fig. 2. Schematic of ensemble learning algorithms.

adopted for the generation of the training data. The time-evolution of the ignition of H_2/air mixtures is obtained from the simulation of constant pressure reactors using Cantera [30]. For simplicity and without loss of generality, the initial pressure is not treated as a variable in our numerical experiment and set to $p = 1.5$ atm; instead, training samples are generated by initializing the temperature T_0 in the range of 1300 and 1500 K, and the fuel-air equivalence ratio ϕ_0 between 0.5 and 2. A total of 330,000 training samples are generated from 800 reactor simulations by discretizing T_0 and ϕ_0 space into 40 and 20 evenly spaced bins, respectively. Benefited from the GOSS and EFB techniques [28], it only takes about 2.8 s for LightGBM to train the current model on a 48-core machine (Intel Xeon Platinum 8275CL @ 3.00 GHz). The trained GBDT model needs about 729 KB of memory for storage.

In this study we focus on presenting the GBDT method as a new approach to the representation of chemical kinetics, thus the initial conditions cover a relative narrow range. However, the scope of initial conditions can be extended according to the applications, and the GBDT model can also be coupled with machine learning techniques such as SOMs and PCA to improve the accuracy over a wide range of scenarios. As machine learning models are data-based, the implementations of ANNs [5,8,12] and GBDTs for chemical kinetics both require the use of a constant time step. In this study we use the same paradigm, but it may be possible to overcome this constraint by introducing sub-iterations to adapt to varying time steps. Also, in the recent study [10], a different framework is proposed where a constant time step treatment is not needed. Mass fractions of species are pre-processed by logarithmic normalization, a procedure that has been commonly used [1,31] to avoid numerical issues due to the radical species concentrations that are highly skewed towards zero.

The database will then be split into two subsets, the training and test datasets, and the accuracy of the trained model can be established if it results in a reasonably small error on the test dataset. That said, a resampling technique called k-fold cross validation (k-fold CV) is suggested to guarantee a more comprehensive assessment [32]. This approach divides the whole database into k equal-size subsets or folds. While one fold is kept as the validation dataset to compute the test error, the other $k-1$ folds will be used for training, and the procedure will be repeated k time as illustrated by Fig. 3.

There are two primary hyper-parameters that determine the structure of a GBDT model, the number of trees (or *rounds*) and the number

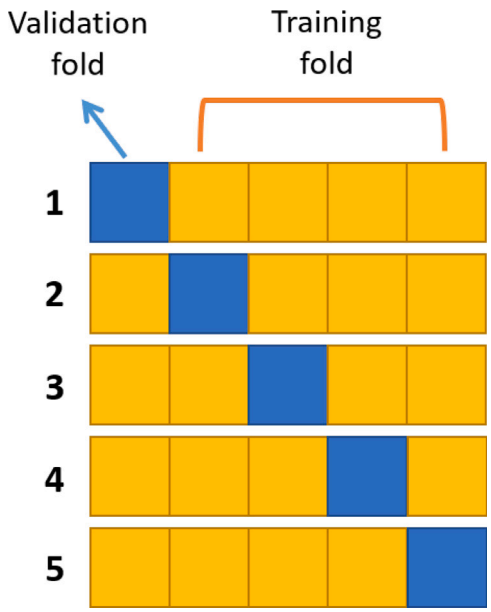


Fig. 3. Scheme of k-fold validation.

of leaves on each tree related to its depth. GBDTs grow exponentially with the maximum depth D_m, that is, the number of leaves equals 2^{D_m} for a level-wise structure. For a leaf-wise GBDT, however, the total number leaves is much lower than 2^{D_m} due to the selection of best nodes to split, a strategy that results in a decision tree with asymmetrical structure. In the present study, the full GBDT model consists of 9 sub-models to account for the 9 species. Since LightGBM adopts a leaf-wise policy, the number of leaves is set to 16 without limiting the maximum depth (will be larger than 4). Thus the GBDTs will stop growing after there are 16 terminal leaves on the tree. This is guided by the recommended setup of the library [28].

The number of decision trees of the sub-models for the species is set to 30, meaning that a total of 30 GBDTs will grow in a sequential manner. For the prediction of temperature, a sub-model is created in which the number of leaves increases to 64 and the total number of GBDTs

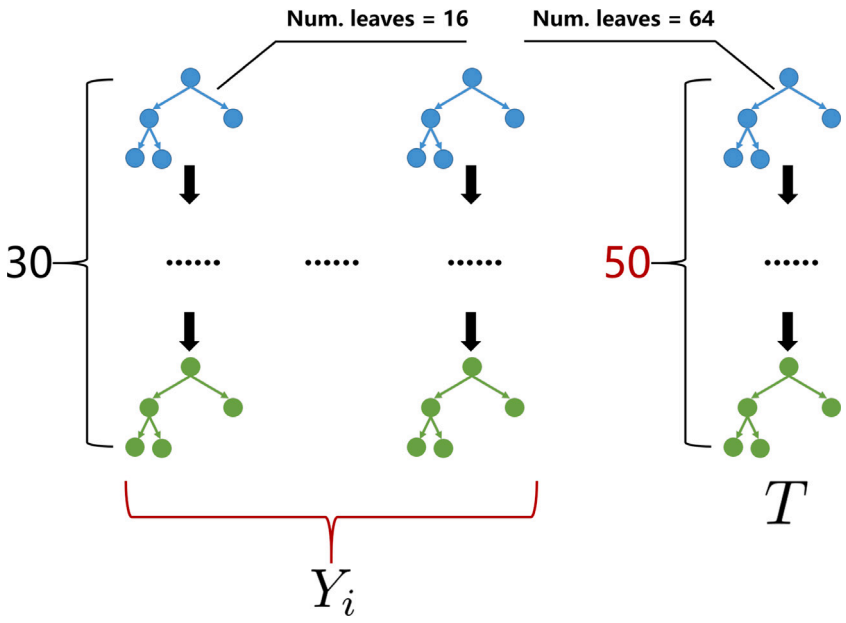


Fig. 4. Ensemble learning framework for the chemical kinetics.

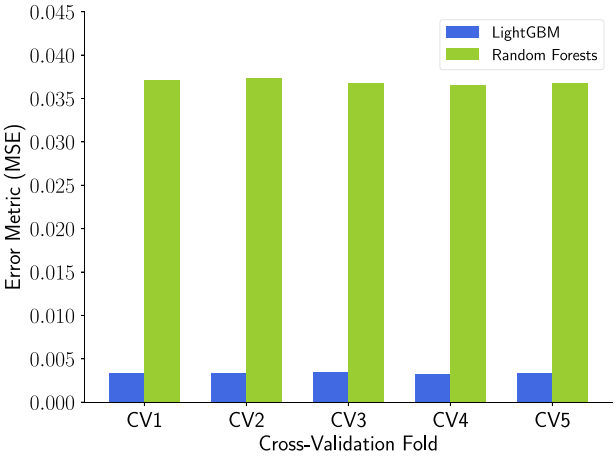


Fig. 5. k-fold validations for bootstrap aggregating and gradient boosting models.

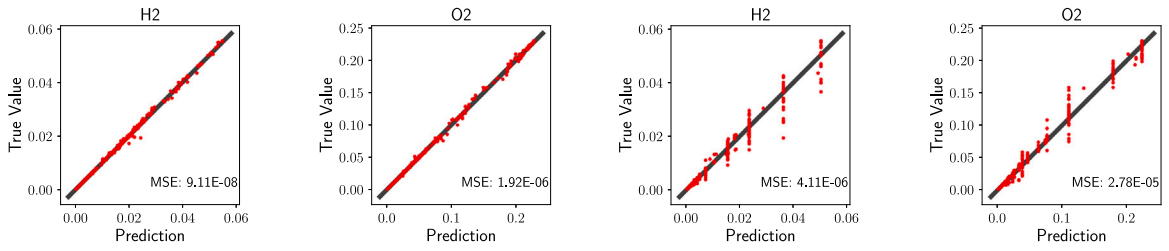
to 50 to ensure high accuracy given the importance of temperature in determining the current thermochemical state (see Fig. 4).

3.2. Prediction and validation

For comparison purposes, the Random Forests library [33] is used to train bootstrap aggregating models, where the hyper-parameters, i.e., the number of trees and leaves, are set to be identical to those of the GBDT model. Fig. 5 shows the cross-validation results between the GBDT and bootstrap aggregating models created by LightGBM and Random Forests, respectively. Note that we repeat the training $k = 5$ times for each method thus there are 5 trained models for each method. The results in Fig. 5 are given by evaluating the kth trained model over the test data in the kth validation fold $\mathrm{CV}_{(k)}$ where the metric is the averaged prediction error of the species compositions (logarithmic scale) of all test samples. As can be seen, the GBDT model outperforms the bootstrap aggregating models by more than an order of magnitude. Besides, the test errors between the 5 validation folds are found to be approximately the same, indicating good generality and robust performance. This is in line with the theoretical analysis that ensemble learning is a machine learning model with low variance error where the predictions will not vary much with regard to the noise of the data [32]. For further demonstration, 100 arbitrary time steps are sampled from the test data and used as model input, and the GBDT model is used to make predictions for the species compositions in the next time step. Fig. 6 shows a scatter plot of the mass fractions predicted by the GBDT and bootstrap aggregating models versus the true values for the major species of H_2 and O_2. The predictions are denoted by the points, and thus the vertical distance from the diagonal line is the measure of deviation. As can be seen, the majority of GBDT predictions fall on the line, whereas a large fraction of predictions by the bootstrap aggregating models are found to deviate considerably from the true values. The bootstrap aggregating models will therefore not be used for further analysis in the remainder of this paper. Furthermore, for the GBDT model, it can be seen from Fig. 7 that the model accuracy improves significantly when the number of trees increases from 20 to 30 (Fig. 6), which is the current setup, but the benefit becomes marginal when this number is set to 40 and the prediction is found to be at the same order of accuracy as that of the current setup.

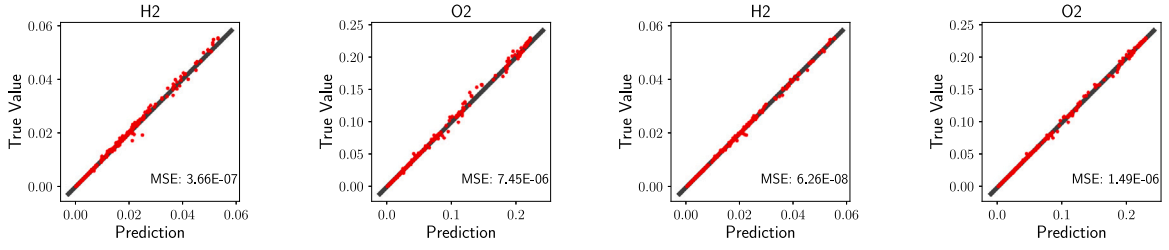
Thus far, the GBDT model is assessed and found to be able to make one time-step predictions of the temperature and species compositions with satisfactory accuracy. For real simulations, however, the objective is an auto-regressive function $[T, p, Y_i]_t \rightarrow [T, p, Y_i]_{t+\Delta t}$ for the time-integration of chemical systems of ODEs. For that, an accurate model should be able to reproduce the full time–history of all species and temperature from the ignition to the final (near equilibrium) conditions. A long-term accuracy assessment is thus conducted by running the GBDT model iteratively and only the initial conditions are given as input. In this case, the predictions of the species composition and temperature at time t are stored, and subsequently used as input for the predictions of a new state at the following time $t + \Delta t$, until the near equilibrium condition has been reached. For demonstration, two initial conditions of the fuel-lean and fuel-rich scenarios are sampled for assessment.

Fig. 8 compares the GBDT predicted time-series with the numerical ODE solutions computed by Cantera under a constant time step of



(a) LigthGBM (b) Random Forests

Fig. 6. Comparison between Random Forests and LightGBM models (Num. Trees = 30).



(a) Num. Tree = 20 (b) Num. Tree = 40

Fig. 7. Sensitivity of the number of trees on GBDT model's accuracy.

Table 1
Mean squared errors and R^2 scores for mass fraction predictions of species.

	$\phi_0 = 0.5, T_0 = 1357\,\mathrm{K}$		$\phi_0 = 1.5, T_0 = 1389\,\mathrm{K}$	
	MSE	R^2	MSE	R^2
H_2	1.49e−07	99.57	4.80e−06	96.79
O_2	4.84e−05	98.21	2.69e−05	99.68
O	9.20e−07	98.10	1.27e−06	91.16
OH	2.25e−07	99.55	3.61e−06	96.41
H_2O	8.75e−06	99.58	2.40e−04	97.31
H	3.34e−08	91.22	3.68e−07	90.64
HO_2	8.78e−12	98.34	3.19e−11	95.55
H_2O_2	4.43e−13	98.52	5.57e−13	94.87

1×10^{-6} s. As can be seen, the GBDT predictions are in good agreement with the numerical solutions except for some deviations near the equilibrium. This is a general issue that also arises when deep ANNs are used [13]. For example, if we let the case of $\phi_0 = 0.5, T_0 = 1357\,\mathrm{K}$ in Fig. 8 continue to run for a longer period of time, it can be seen that the predictions towards the equilibrium turn flat and the minor changes of the mass fractions and temperature cannot be well captured (see Fig. 9). This issue could get worse if a smaller time step is considered, e.g., 5×10^{-7} s or 1×10^{-7} s, and thus the capability of the GBDTs (e.g., the size or depth of trees) needs to be enhanced. It could also be mitigated by adding a correction step to the prediction to enhance long-term stability. For example, Wan et al. [8] added a secondary ANN to their main ANN model, which was specifically trained on a subset range of the data and then switched to work when the reacting rates of flames were slow. Likewise, in real applications sub-models of GBDTs can be added and activated based on the progress variable. It is also noted that the prediction for certain species, such as H_2O, is not yet optimal, thus increasing the numbers of leaves and trees of the associated sub-model may be needed. Besides, the accuracy of the GBDT model needs to be further improved for the cases with initial conditions on the bounds of the parameter ranges, i.e., T_0 = 1300 K or 1500 K, and ϕ_0 = 0.5 or 2.0. The mean squared error and R^2 score for the mass fraction prediction of each species are summarized in Table 1. For the fuel-lean and fuel-rich cases, the time–history yields overall mean squared errors of 1.5×10^{-5} and 5.0×10^{-6}, respectively, between the predicted time series and the numerical solutions for species compositions, and the temperature deviates by 1%–3% from the correct solutions.

For the scenarios of low temperature conditions (LTCs), such as T_0 = 800 K–900 K, additional treatments may be required. Due to the multi-scale characteristics of species concentrations and reacting rates, it is very challenge to use a global GBDT model to predict both the long ignition delay period and the rapid exothermic reactions afterwards. In this case, the GBDT model should be implemented with, for example, the SOM technique [4–6] to divide the composition space into sub-domains or with the PCA [17,34,35] to map the high-dimensional composition space to lower-dimensional manifolds. On the other hand, for the LTC cases, a much larger time step of integration is allowed during the long ignition delay period (Fig. 10) (and towards the equilibrium), meaning that the stiffness of the ODEs is much less serious and numerical ODE solvers can be switched on during these periods as a supplement to the GBDT model.

3.3. Code integration and performance

As we aim to use the GBDT model to avoid the need of direct integration of the stiff ODE systems, the speed-up performance is of great concern when the models are implemented to CFD codes. The LightGBM library, as the name implies, is a lightweight library that gives ensemble learning an inbuilt advantage over deep learning. The GBDT model can be readily exported as dependency-free C/C++ codes with the help of the Model 2 Code Generator (m2cgen) [36]. The latter can then work as a stand-alone prediction routine without the need to install the LightGBM library. Readers are referred to the supplementary

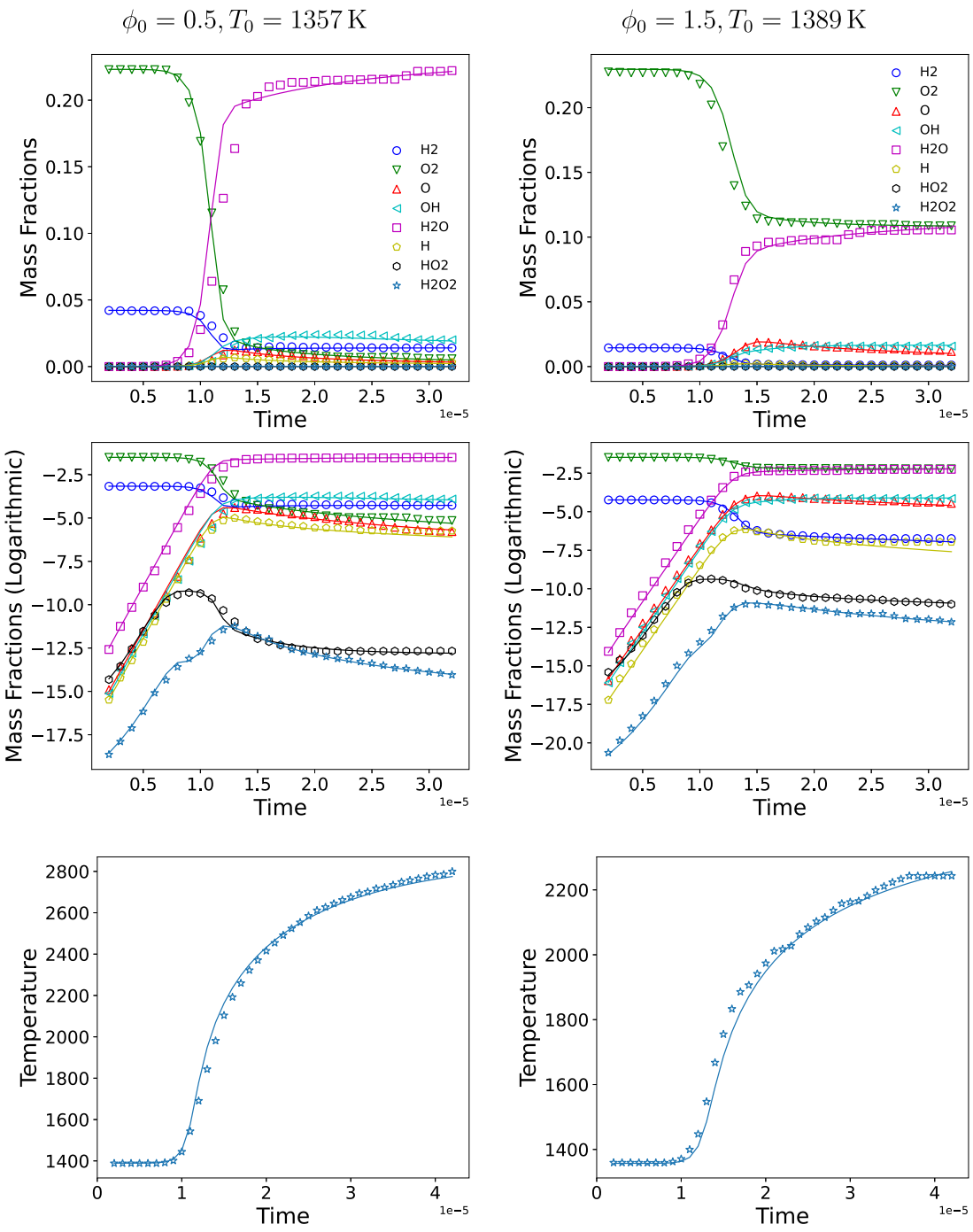


Fig. 8. Time-series prediction of the ignition of H_2/air mixtures (markers for the GBDT predictions and solid lines for the numerical ODE solutions).

material of our previous study [23] where examples and a user manual are offered. That said, the trained GBDT model is converted into C++ codes and compared with the numerical solver in OpenFOAM [37] for the simulation of a one-dimensional freely propagating flame. The result suggests that the GBDT model is on average one order of magnitude faster than the numerical solver (Fig. 11). These numbers are obtained without further optimization and therefore demonstrate the potential of the GBDT model to reduce the computational cost of solving the stiff ODE systems in reactive flow simulations. The prediction of the fuel distribution is also presented, but this test case is mainly used to evaluate the computation efficiency of the GBDT model; for better accuracy the model needs to be further optimized.

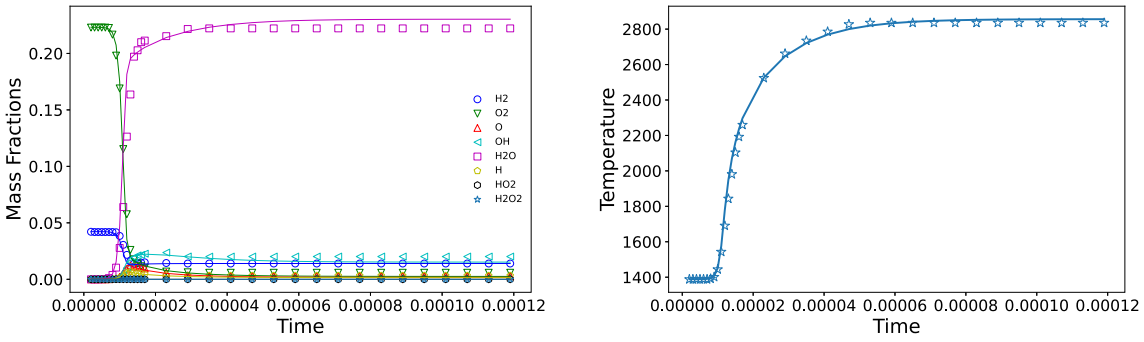


Fig. 9. Behavior of GBDT predictions towards the equilibrium ($\phi_0 = 0.5, T_0 = 1357\,\mathrm{K}$). The plots draw every 6th points for clarity.

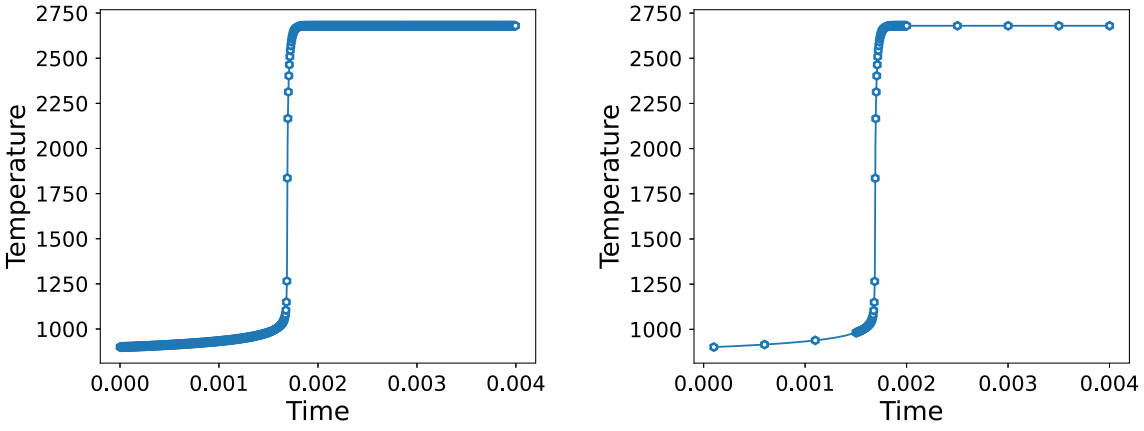


Fig. 10. Ignition of LTC cases (T_0 = 900 K) with fixed (1×10^{-6} s) and varying (1×10^{-4} s and 1×10^{-6} s) time steps (Cantera solutions). H_2O_2 with an initial mass fraction of 0.046 is seeded to accelerate ignition.

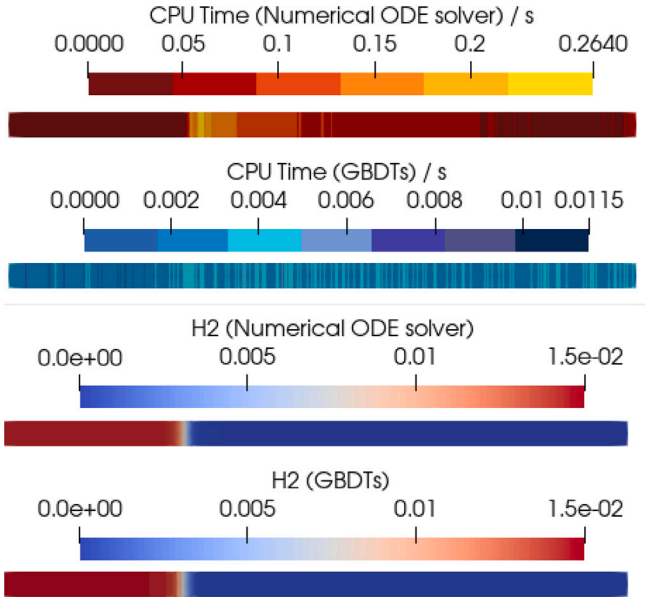


Fig. 11. Speed-up performance of the GBDT model in simulating a one-dimensional freely propagating flame.

4. Conclusions

Ensemble learning is introduced as an efficient machine learning method to reduce the computational cost of solving the stiff ODE systems that are characteristic for many simulations of reactive flows with detailed chemistry. The GBDT model is trained on a database of H_2 oxidation with a detailed H_2/O_2 mechanism using the LightGBM library. A comparative study shows that the gradient boosting algorithm performed by LightGBM exhibits superior performance to the bootstrap aggregating algorithm typified by Random Forests. The GBDT model is used to reproduce the entire time–history profile of hydrogen combustion from ignition to equilibrium composition. Here the GBDT model makes predictions iteratively to generate a full time series of a reacting H_2/O_2 mixture using initial conditions as the only input for the trained model. The results are compared against the solutions computed by a standard solver. It is found that the GBDT model predicts the species compositions and temperature with reasonable accuracy that is also comparable to published deep learning results using ANNs. The GBDT model implementation into C++ codes demonstrates a reduction of the computational cost of solving the ODEs by one order of magnitude when compared to a conventional (optimized) ODE solver. There is no need to couple the LightGBM library with the CFD code which makes the method promising alternative to ANNs when in search of machine learning methods for the reduction of computational cost for the integration of a stiff ODE system.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

S. Yao and W. Zhang acknowledge the support from the Ningbo Science and Technology Innovation Project No. 2020Z071.

References

[1] Blasco JA, Fueyo N, Dopazo C, Ballester J. Modelling the temporal evolution of a reduced combustion chemical system with an artificial neural network. Combust Flame 1998;113(1–2):38–52.
[2] Abadi M, Agarwal A, Barham P, Brevdo E, Chen Z, Citro C, et al. TensorFlow: Large-scale machine learning on heterogeneous systems. 2015, Software available from tensorflow.org.
[3] Ihme M, Schmitt C, Pitsch H. Optimal artificial neural networks and tabulation methods for chemistry representation in LES of a bluff-body swirl-stabilized flame. Proc Combust Inst 2009;32(1):1527–35.
[4] Chatzopoulos AK, Rigopoulos S. A chemistry tabulation approach via rate-controlled constrained equilibrium (RCCE) and artificial neural networks (ANNs), with application to turbulent non-premixed CH4/H2/N2 flames. Proc Combust Inst 2013;34(1):1465–73.
[5] Franke LLC, Chatzopoulos AK, Rigopoulos S. Tabulation of combustion chemistry via Artificial Neural Networks (ANNs): Methodology and application to LES-PDF simulation of Sydney Flame L. Combust Flame 2017;185:245–60.
[6] Ranade R, Li G, Li S, Echekki T. An efficient machine-learning approach for PDF tabulation in turbulent combustion closure. Combust Sci Technol 2019;1–20.
[7] Owoyele O, Kundu P, Ameen MM, Echekki T, Som S. Application of deep artificial neural networks to multi-dimensional flamelet libraries and spray flames. Int J Engine Res 2020;21:151–68.
[8] Wan K, Barnaud C, Vervisch L, Domingo P. Chemistry reduction using machine learning trained from non-premixed micro-mixing modeling: Application to DNS of a syngas turbulent oxy-flame with side-wall effects. Combust Flame 2020;220:119–29.
[9] Owoyele O, Kundu P, Pal P. Efficient bifurcation and tabulation of multi-dimensional combustion manifolds using deep mixture of experts: An a priori study. Proc Combust Inst 2021;38(4):5889–96.
[10] Owoyele O, Pal P. ChemNODE: A neural ordinary differential equations framework for efficient chemical kinetic solvers. Energy AI 2022;7:100118.
[11] Seltz A, Domingo P, Vervisch L, Nikolaou ZM. Direct mapping from LES resolved scales to filtered-flame generated manifolds using convolutional neural networks. Combust Flame 2019;210:71–82.
[12] Wan K, Barnaud C, Vervisch L, Domingo P. Machine learning for detailed chemistry reduction in DNS of a syngas turbulent oxy-flame with side-wall effects. Proc Combust Inst 2021;38(2):2825–33.
[13] Ji W, Deng S. KiNet: A deep neural network representation of chemical kinetics. 2021, arXiv preprint arXiv:2108.00455.
[14] Zhang T, Yi Y, Xu Y, Chen ZX, Zhang Y, Xu Z-QJ, et al. A multi-scale sampling method for accurate and robust deep neural network to predict combustion chemical kinetics. 2022, arXiv preprint arXiv:2201.03549.
[15] LeCun Y, Boser B, Denker JS, Henderson D, Howard RE, Hubbard W, et al. Backpropagation applied to handwritten zip code recognition. Neural Comput 1989;1(4):541–51.
[16] He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016, p. 770–8.
[17] Dalakoti DK, Wehrfritz A, Savard B, Day MS, Bell JB, Hawkes ER. An a priori evaluation of a principal component and artificial neural network based combustion model in diesel engine conditions. Proc Combust Inst 2021;38(2):2701–9.
[18] Gitushi KM, Ranade R, Echekki T. Investigation of deep learning methods for efficient high-fidelity simulations in turbulent combustion. Combust Flame 2022;236:111814.
[19] Kohonen T. The self-organizing map. Proc IEEE 1990;78(9):1464–80.
[20] Chung WT, Mishra AA, Perakis N, Ihme M. Data-assisted combustion simulations with dynamic submodel assignment using random forests. Combust Flame 2021;227:172–85.
[21] Peters N. Turbulent Combustion. Cambridge University Press; 2000.
[22] de Frahan MTH, Yellapantula S, King R, Day MS, Grout RW. Deep learning for presumed probability density function models. Combust Flame 2019;208:436–50.
[23] Yao S, Kronenburg A, Stein O. Efficient modeling of the filtered density function in turbulent sprays using ensemble learning. Combust Flame 2021;111722.
[24] Hastie T, Tibshirani R, Friedman JH, Friedman JH. The elements of statistical learning: data mining, inference, and prediction. vol. 2, Springer; 2009.
[25] Subramaniam S, Pope SB. A mixing model for turbulent reactive flows based on Euclidean minimum spanning trees. Combust Flame 1998;115:487–514.
[26] Breiman L. Random forests. Mach Learn 2001;45(1):5–32.
[27] Chen T, Guestrin C. XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining. 2016, p. 785–94.
[28] Ke G, Meng Q, Finley T, Wang T, Chen W, Ma W, Ye Q, Liu T-Y. LightGBM: a highly efficient gradient boosting decision tree. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. 2017, p. 3149–57.
[29] Li J, Zhao Z, Kazakov A, Dryer FL. An updated comprehensive kinetic model of hydrogen combustion. Int J Chem Kinet 2004;36(10):566–75.
[30] Goodwin DG, Moffat HK, Speth RL. Cantera: An object-oriented software toolkit for chemical kinetics, thermodynamics, and transport processes. vol. 124, Pasadena, CA: Caltech; 2009.
[31] Sharma AJ, Johnson RF, Kessler DA, Moses A. Deep learning for scalable chemical kinetics. In: AIAA scitech 2020 forum. 2020, p. 0181.
[32] James G, Witten D, Hastie T, Tibshirani R. An introduction to statistical learning. vol. 112, Springer; 2013.
[33] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, et al. Scikit-learn: Machine learning in Python. J Mach Learn Res 2011;12:2825–30.
[34] Sutherland JC, Parente A. Combustion modeling using principal component analysis. Proc Combust Inst 2009;32 I(1):1563–70.
[35] Malik MR, Obando Vega P, Coussement A, Parente A. Combustion modeling using Principal Component Analysis: A posteriori validation on Sandia flames D, E and F. Proc Combust Inst 2021;38(2):2635–43.
[36] m2cgen (model 2 code generator). 2020, https://github.com/BayesWitnesses/m2cgen.
[37] Weller HG, Tabor G, Jasak H, Fureby C. A tensorial approach to computational continuum mechanis using object-oriented techniques. Comput Phys 1998;12:620.