Fire Safety Journal 114 (2020) 102991

Contents lists available at ScienceDirect

Fire Safety Journal

journal homepage: http://www.elsevier.com/locate/firesaf

Using machine learning in physics-based simulation of fire

B.Y. Lattimer [a,*], J.L. Hodges [a], A.M. Lattimer [b]

[a] Jensen Hughes, RDT&E, Blacksburg, VA, USA
[b] Socially Determined, Blacksburg, VA, USA

ARTICLE INFO

Keywords:
Machine learning
Fire models
Wildland fires
Building fires
Real-time

ABSTRACT

There is a current need to provide rapid, high fidelity predictions of fires to support hazard/risk assessments, use sparse data to understand conditions, and develop mitigation strategies. Machine learning is one approach that has been used to provide rapid predictions based on large amounts of data in business, robotics, and image analysis; however, there have been limited applications to support physics-based or science applications. This paper provides a general overview of machine learning with details on specific techniques being explored for performing low-cost, high fidelity fire predictions. Examples of using both dimensionality reduction (reduced-order models) and deep learning with neural networks are provided. When compared with CFD results, these initial studies show that machine learning can provide full-field predictions 2–3 orders of magnitude faster than CFD simulations. Further work is needed to improve machine learning accuracy and extend these models to more general scenarios.

1. Introduction

Predicting the detailed behavior of fires is a complex process that involves fluid mechanics, heat transfer, combustion, and interaction with surroundings. As a result, computational fluid dynamics (CFD) models to predict the full-field conditions that result from fires are computationally expensive. The current state-of-the-art models, such as Fire Dynamics Simulator (FDS) [1] and FireFOAM [2], can provide detailed predictions over smaller to moderate domain sizes (up to 100s of meters). In several applications (wildland fires, mines, buildings, ships), there is a need to predict detailed conditions over very large domains (10s of kilometers) to accurately assess the hazards, perform risk assessments, and develop mitigation strategies. In addition, as the internet of things (IoT) becomes more prevalent, low cost models will be needed to use sparse data being collected to provide situational awareness and hazard evaluation. This paper contains an overview of the use of some machine learning techniques that are being explored to provide low cost, detailed simulation results of fires.

Predicting fire behavior over large domains has historically been limited to simplified physics models such as network (or zone) models. These network models solve for average conditions (temperature, velocity) over a region making them more computationally efficient than high fidelity CFD models. The change in the conditions in the region are determined using conservation of mass, momentum, species, and energy. One example of this type of model for fire applications is CFAST [3], where building rooms are represented as one or two-layer environments. However, there is a limitation to the number of rooms that can be predicted using this model. Fire and Smoke Simulator (FSSIM) [4] is another type of network model where rooms are represented using one or two-layer environments, but it does not have a limitation on the number of rooms that can be simulated. The downside to these models is that they only provide average conditions in the room (e.g., one-layer model is the average conditions in the entire room while two-layer models are average conditions in the upper part of the room and average conditions in the lower part of the room).

The most common method to date for predicting detailed conditions in large domain problems is to apply a hybrid modeling approach. In this approach, a network model is used to predict the fire conditions over the entire domain and a CFD model is subsequently used to predict a small sub-region of the domain where large spatial variations are expected or are of interest. This approach has been applied to tunnel applications [5–8] as well as buildings [8]. While this typically provides the information needed for quantifying the hazard, the computational expense is still high since a CFD simulation must be conducted. As a result, this methodology is too slow to perform multiple simulations to support risk assessments or evaluate potential mitigation strategies.

An alternative approach is to use machine learning to provide rapid predictions of high fidelity results based on an input. Machine learning

* Corresponding author.
E-mail address: lattimer@vt.edu (B.Y. Lattimer).

https://doi.org/10.1016/j.firesaf.2020.102991
Received 16 July 2019; Received in revised form 24 January 2020; Accepted 13 March 2020
Available online 14 March 2020

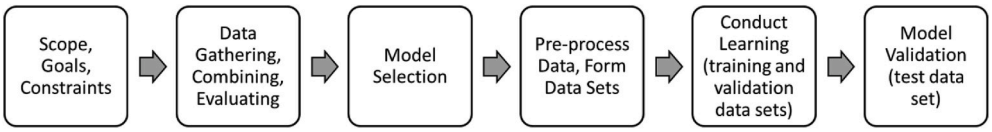


Fig. 1. Approach to machine learning design.

has been broadly used in business, robotics, and image processing; however, only select machine learning techniques have been used in physics-based applications. The two methods that have been most commonly applied are dimensionality reduction (reduced-order models) and artificial neural networks. This paper provides a brief overview of machine learning followed by example applications of using machine learning techniques for predicting fire conditions. The ability of these techniques to predict full-field fire conditions is provided along with the existing challenges of using these types of approaches.

2. Machine learning overview

Machine learning is a broad field where large amounts of data are used to predict an outcome or extract insight from the data. General overviews of the topic can be found elsewhere [9]. Within this framework, there is a model that has parameters which need to be learned for the model to produce an output based on input data. Machine learning uses large amounts of data to determine the parameters so that the model predicts something (predictive), is able to gain insight from the data (descriptive), or both. There are three general categories of machine learning: supervised, unsupervised, and reinforcement learning. Supervised learning uses labeled data to develop a relationship between input data and a desired output. Unsupervised learning uses unlabeled data to determine how the input data relates to the output. In reinforcement learning, the learning is developed by providing rewards or punishments related to the previous results from the process. The research described in this paper is based on both supervised and unsupervised machine learning.

The general approach to machine learning design is shown in Fig. 1 and is a generalized approach of that recommended by Hodges [10]. The first step is to define the scope, goals, and constraints of the problem. This includes the desired inputs and outputs from the machine learning model, the expected limits of each input and output, and the topology of the data (such as individual point in space or full-field, individual point in time, or resolved). Data is then gathered and combined to support the machine learning. In physics applications, the data may come from the literature, historic data, new experiments, point measurements in a system, or generated using simulations. The available data is evaluated with statistical techniques to develop a high-level understanding of its form. Following this analysis, the appropriate machine learning model is selected. The optimal model for a specific problem will be dependent on its scope, goals, and constraints, the form of the input and outputs, as well as the availability of data. With the model selected, the data is then pre-processed to enable input into the machine learning. In addition, data is separated into three data sets with similar statistical properties: training, validation, and testing sets. The training data is used to optimize the model parameters while the validation data set is used periodically during the training process to determine how well the model generalizes to unseen scenarios (i.e., ensure no overfitting with the training data set). Modeling parameters are tuned until the model predictions on both the training and validation data sets meet the desired performance criteria. Once the model parameters have been optimized through the learning process, the model is then evaluated using the testing data set that was not included in the learning. As a result, applying the testing data set to the model provides an unbiased estimate on the model generalization.

Machine learning for physics-based simulations is a relatively new area of study. In supervised learning, neural networks have been broadly used to support physics-based predictions. Most of these applications have focused on predicting single point values; however, recent work has been exploring the use of generative models to provide two dimensional predictions. In unsupervised learning, dimensionality reduction using principal component analysis has been widely used in the field of fluid mechanics to understand flow structure [11] and more recently to provide computational efficient results [12]. The following sections provide an overview of some previous work in each of these areas and the application of these techniques to predicting fire behavior.

3. Dimensionality reduction

3.1. Overview

Dimensionality reduction using principal component analysis, also known as reduced-order modeling (ROM), has been explored for predicting detailed fire dynamics with less computational expense. In this approach, a new set of mathematical equations is developed to project full-order equations into a reduced-order space. This approach is unsupervised machine learning since the data does not need to be labeled to learn the projection. This has been done for simple buoyancy driven plumes without reactions [13] and wildland fire spread models [14,15]. With these implementations, reduced-order models were able to provide prediction times that were 2–3 orders of magnitude less than the full-order models (i.e. CFD model). An overview of the results for the buoyancy driven 2D plume are provided here to demonstrate the technique and the potential for using this approach in fire related problems. Note that this preliminary work does not include the detailed effects of reactions needed for fires, which requires additional equations to include into the framework.

The model reduction technique used in this work consisted of creating a proper orthogonal decomposition (POD) basis by truncating the left singular vectors generated using a singular value decomposition (SVD) of a data snapshot matrix. This POD basis is then used to project the governing PDEs onto a low-dimensional subspace. The result is a small system of ordinary differential equations (ODE) that can replace the complexity of the full-order model while retaining nearly the same accequations (ODE) that can replace the complexity of the full-order model while retaining nearly the same accuracy as long as the physics of the system remain “close” to the benchmark simulations. There are essentially five main steps that must be accomplished in order to create a ROM POD:

1. Capture data snapshots of the full-order model (FOM).
2. Build POD basis functions from the snapshots.
3. Compute and store the intermediate inner product matrices required to create the ROM ODE to be solved.
4. Load the inner product matrices and assemble the ROM ODE.
5. Solve the time-dependent ROM ODE using a standard ODE solver.

It is worth noting that Steps 1–3 above are considered off-line costs. Once accomplished there is no need to repeat them unless the size of the ROM is changed or the data snapshots are altered. Steps 4–5 are the real-time costs for solving the problem. It should be emphasized the diffusion constants and/or the initial conditions can be adjusted without having to repeat Steps 1–3 as long as the initial condition remains relatively close to the initial condition used to generate the data snapshots.

Table 1
Model constants and solution times for different ROM sizes, r.

r	ν	α	Solution time (s)
20	1.0e-02	1.0e-02	3.7
50	9.0e-03	9.0e-03	14.4
100	3.0e-03	3.0e-03	72.3

3.2. Example application

The focus of this effort was to create a ROM of the mass, momentum, and energy equations associated with buoyancy driven plumes from fires. FDS was used to generate the full-order fire plume model. The 2D domain was 1.0 m wide and 2.0 m high which was open to the atmosphere on the sides and top. The base of the domain was a solid gypsum surface with a 0.2 m wide methane burner centered on the base. The burner was set to release methane at a rate to generate a 40 kW fire.

The system was discretized to 50 elements in the x-direction (horizontal) and 100 in the y-direction (vertical). The system was simulated from [0,20] seconds. This gives a spatial mesh of nodal values evenly spaced every 0.02 m in both directions, i.e. 51×101 nodes, resulting in a FOM size of $n = 5151$. Five hundred evenly spaced snapshots were captured over the time domain $t = [0,20]$ to represent the entire full-order plume fire model. To produce a ROM from this data, the PDEs in Eqns. (1)–(3) were used as the model for conservation of momentum, mass, and energy:

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{u}\cdot\nabla\mathbf{u} - \nabla p + \nu\nabla^2\mathbf{u} - \beta\mathbf{g}(T - T_\infty), \tag{1}$$

$$0 = \nabla\cdot\mathbf{u}, \tag{2}$$

$$\frac{\partial T}{\partial t} = -\mathbf{u}\cdot\nabla T + \alpha\nabla^2 T. \tag{3}$$

These are the same equations as the general fire model equations used in FDS with the chemical reaction term removed. With these equations, the term β was taken to be temperature dependent ($\beta = 1/T$) to capture buoyancy aspects of the plume. The impact of this change over using a constant value for β is that the body force term can no longer be handled by precomputing the inner product. As a result, the term is computed by lifting the temperature to the full state space and then projecting the result back down reduced state space. The building and execution of the ROM was performed in MATLAB.

From the snapshots, r POD basis modes were determined for the u horizontal velocity, v vertical velocity, and temperature T of size r. In addition, diffusion constants ν and α needed to be optimized based on the ROM size to achieve the best results. These constants were developed to provide the best agreement between the temperature and velocity solutions. Adding combustion to the model would likely remove the need to change these diffusion constants. Table 1 contains the ROM model constants and simulation times in MATLAB. FOM simulation times were approximately 550 s. As seen in the table, decreasing the size of the ROM decreases the simulation.

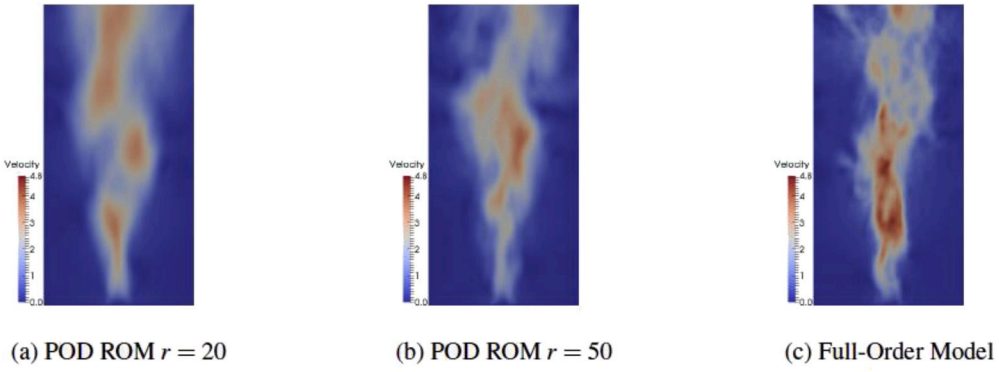


Fig. 2. Velocity predicted using different size ROMs and the full-order model (FOM).

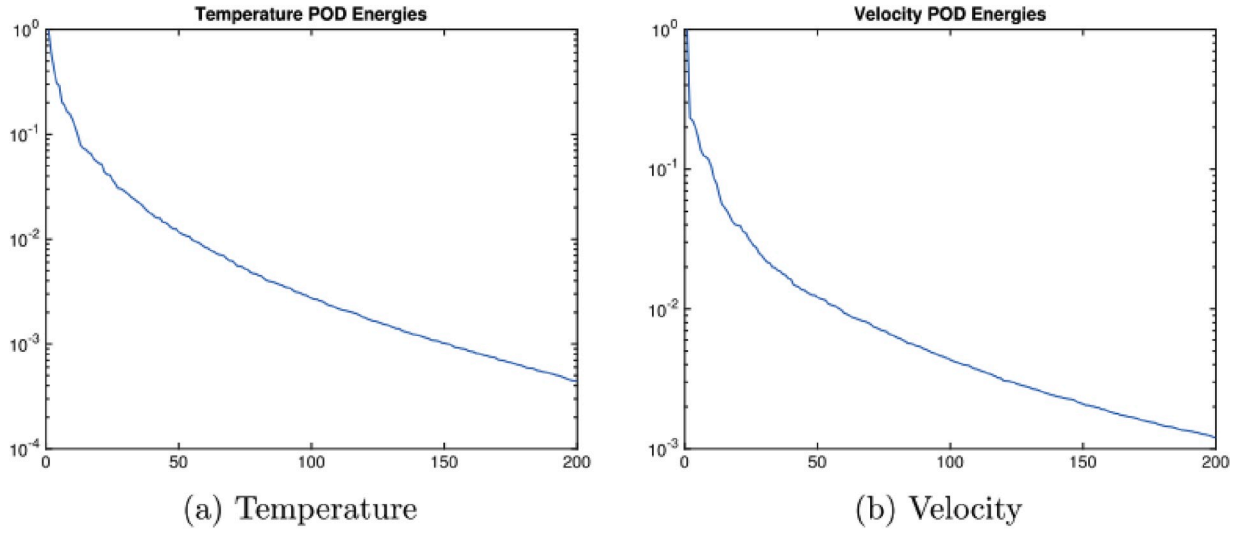


Fig. 3. Energy of different modes for temperature and velocity.

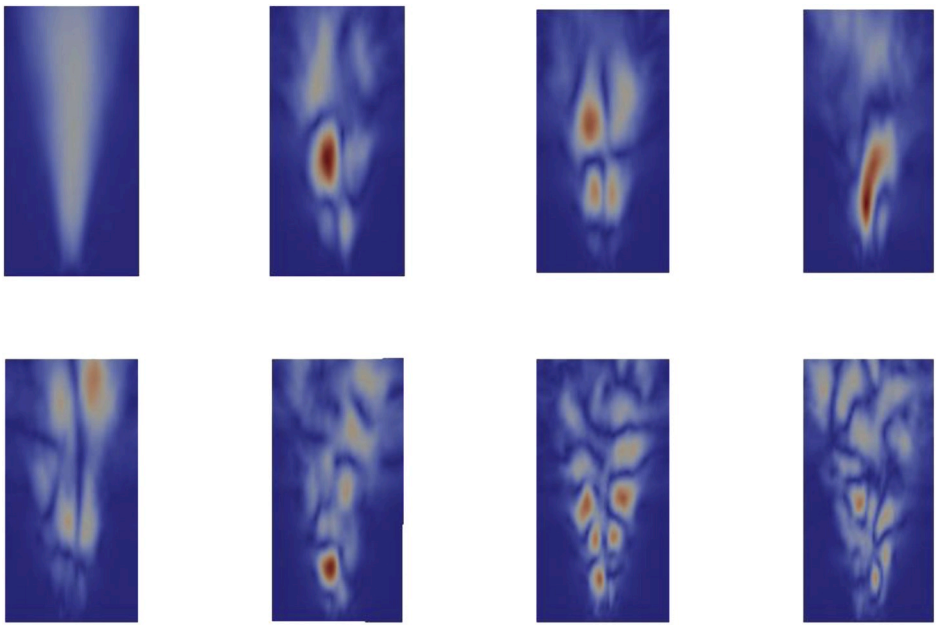


Fig. 4. Velocity ROM Modes 1, 2, 3, 4, 5, 10, 15 and 20 (from right to left, top to bottom) showing the different structure added to the flow by each mode.

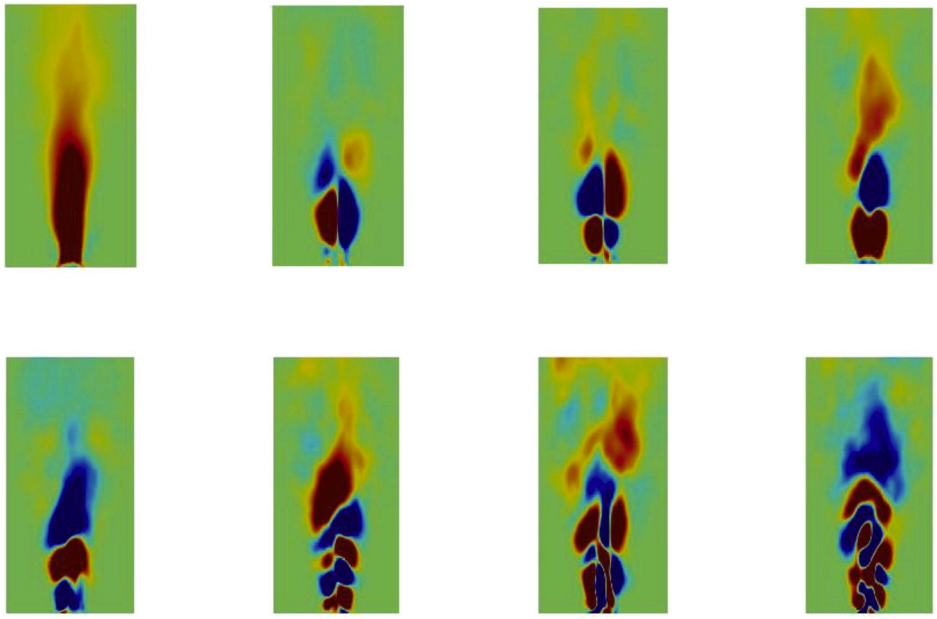


Fig. 5. Temperature ROM Modes 1, 2, 3, 4, 5, 10, 15 and 20 (from right to left, top to bottom) showing the different structure added to the flow by each mode.

Several tests were run to compare the ROM to the FOM generated by FDS. Results of a velocity comparison are provided in Fig. 2 for two different size ROMs. As seen in the comparison, a ROM of $r = 20$ provides a reasonable representation of the global flow field with a speed up of 150X. Increasing the ROM size to $r = 50$ results in some finer scale details being captured, but the computational speed up is reduced to 40X. In general, both ROMs are able to provide a rapid estimation of the flow field.

There is no optimal number of basis modes, r, to use in the solution. As previously shown, the more modes added will tend to add more accuracy and detail to the model results. As a result, a sufficient number of modes needs to be selected to capture the essence of the flow field while minimizing the computational cost. One way to assess this is to look at the magnitude of contribution to the flow (energy) for each mode as shown in Fig. 3. As seen in this figure, the majority of the energy is captured in the initial 25–50 modes for both temperature and velocity. Each problem will have a different distribution in mode energy; therefore, the mode energy should be determined for each different problem to assess the appropriate size of the ROM.

Another advantage of using POD is that it breaks up the solution into a series of modes that each contain information about the flow field structure. When all of these modes are added together, they generate the overall flow field. For example, Fig. 4 and Fig. 5 contain select velocity and temperature modes from the plume ROM shown previously. The first mode provides the average flow conditions in the plume region and outside the plume region. Each additional mode adds different levels of coherent structure with lower modes providing more large-scale structure and the higher modes giving the smaller scale detail. This is in part

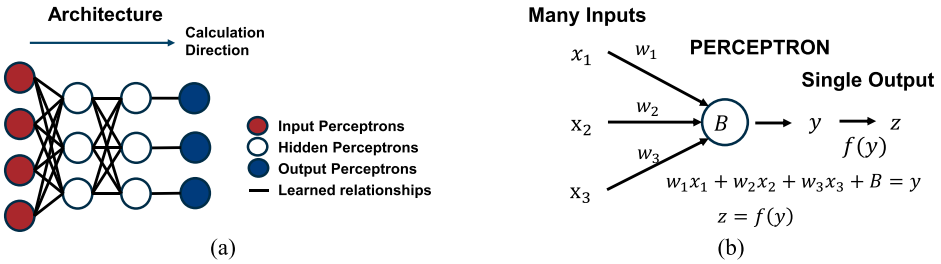


Fig. 6. a) Artificial neural network general architecture form and b) construct details of a single perceptron.

why POD is frequently used in analyzing flow fields measured using particle image velocimetry, since it can separate out different flow field structure to better understand the flow.

Improving the results of the ROM for fires requires reconfiguring the ROM to include the species and combustion aspects of the model, making it more similar to the equations solved by FDS. In addition, this will require the handling of several nonlinear terms. Though this is possible, this essentially results in the generation of a new set of governing equations to solve. This complication makes the use of reduced-order models for fire applications to be an area that involves significantly more mathematical and numerical analysis study before being able to be used for reacting flow problems.

4. Deep learning

4.1. Overview

Artificial neural networks (ANN) or multilayer perceptrons are a type of model that uses a collection of interconnected nodes (perceptrons) to represent the relationship between an input and an output. As shown in Fig. 6a, ANNs consist of an input layer, a series of hidden layers, and an output layer. If there are more than two hidden layers, this is considered deep learning.

Each perceptron takes in multiple inputs and then produces a single output, see Fig. 6b. Inputs are weighted results from other perceptions or provided to the ANN (input layer). Each perceptron also has a bias. As a result, each perceptron forms a linear equation that relates the inputs and bias to the output. To add nonlinearity to the system, an activation function is applied onto the output so that the system can learn nonlinear relationships. Some common activation functions include a step function, sigmoid, tanh, rectified linear unit, or leaky rectified linear unit. Assembling the entire network results in a series of linear equations. Machine learning is used to determine the weights and biases so that the input predicts the output.

Researchers have successfully applied data-driven artificial neural network (ANN) approaches to predict physics applications including wildland fire spread [16–20], storm surge [21–24], flood inundation [25–27], climate modeling [28,29], remote sensing [30,31], and power generation [32,33]. Although ANNs have been used in physics-based applications, the predictions have primarily been limited to estimates of a single quantity at a single point. Generative modeling is a new field of research in the machine learning community which has primarily been focused on constructing RGB images. Recently, these approaches have been adapted to CFD applications to study wake fluid flow [34–36], turbulence modeling [37,38], and thermal hydraulics [39]. Some studies has attempted to predict conditions of a flow field based on computational cell boundary conditions [40–42]; however, this has not been expanded to perform full-field predictions with changes in overall boundary conditions.

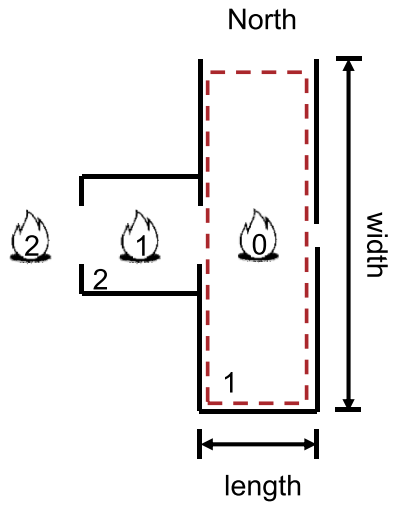
Recently, ANNs have been used to predict full-field fire conditions in building fires (gas temperatures, velocities) and wildland fires (fire spread maps) [10]. This is different than previous work in that the ANN was developed using convolutional neural networks to learn the flux relationships from kernels based on the full-field CFD model training data. An example application of this is provided below predicting fire conditions inside of mines to support ventilation design and risk assessment due to the occurrence of fires [10].

4.2. Example application

Predicting the flow and developing conditions inside of a mine during a fire requires a model that is capable of performing predictions over kilometers. This is typically done with a network model, which provides volume average flow, temperature, and species concentrations. However, the spatially resolved flow details which may be important to assess the hazard are not determined and are computationally expensive to do with a CFD model. Since most mines have a network model built for their system that determines volumetrically average flow conditions, machine learning was used to determine the spatially resolved conditions (velocity, temperature) based on the volumetrically average flow field conditions, geometry, and fire description. The input data for the machine learning model [43] were the volumetrically average conditions at points in the geometry while the desired output was a two-dimensional slice of the spatially resolved conditions (velocities, temperature). Due to the output being a two-dimensional field, an up-convolutional neural network model architecture was identified to be able to use discrete input values to learn the two-dimensional flow field conditions. This approach was inspired by the work performed by Dosovitskiy et al. [44,45], where these types of ANNs were used to predict unseen views of objects based on high level descriptors of the object (style, orientation, color, view, etc.). In the fire application, these input flow descriptors are used along with kernels (filters) to learn fluxes produced in the overall flow field conditions that will exist based on the geometry and fire scenario.

In previous work, an ANN architecture was developed to predict the flow field conditions in a structure fire [43]. In this work, a simplified multi-compartment geometry was simulated in FDS to develop a training and test set for the ANN machine learning. The training was performed on over 1000 simulation results and included zero velocity results to improve small fire results. Select output on the gas temperature and velocity were provided to show the predictive capability of the ANN. Results were then used to predict a more complex multi-compartment geometry based on the simplified geometry training. In the example below, the same ANN architecture is used along with the simplified multi-compartment geometry as in the previous work. However, results from only 220 simulations were used to train and test the ANN machine learning model. The predictive capability of this ANN model using fewer data in the training is demonstrated with a wider range in input parameters and more test case comparison results.

Data sets to perform and evaluate the machine learning were developed using 220 FDS simulations. In order to generate a sufficient amount of data to support the machine learning, a simplified geometry was constructed that would support running hundreds of simulations. An initial assessment of the approach was developed based on a two-room fire scenario shown in Fig. 7. This is the same geometry used for training in previous work [43]; however, few simulations were



Room Geometry	
Width	2.5-10.0 m
Length	2.5-10.0 m
Height	2.5 m
Each Wall (North, East, South, West, West-West)	
Existence of Ventilation	True/False
Ventilation Size	0-2
0	0.625x1.9m
1	1.250x1.9m
2	Full Extent
Source	
Location	0-2
0	Fire Room 1
1	Fire Room 2
2	2-Layer Forced Ventilation
Intensity	
if Fire, Heat Release Rate	50-1000 kW
if 2-Layer Forced Ventilation	
Temperature	500-1000 C
Velocity	1-10 m/s
Interface Height	0-1.9m

(b)

Fig. 7. The two-room configuration used to generate data sets with FDS.

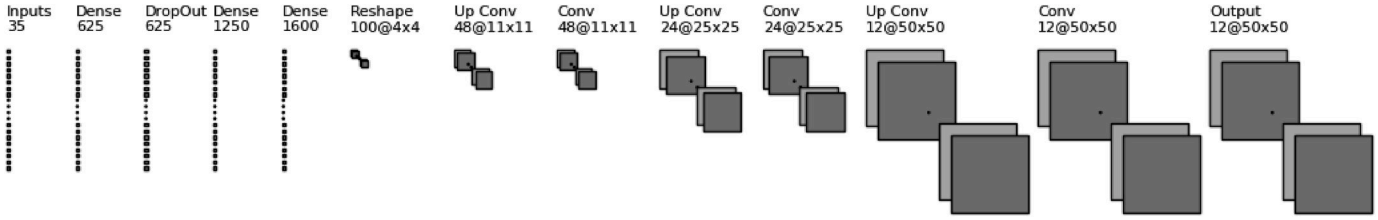


Fig. 8. Up-convolutional artificial neural network (ANN) used to predict fire spatial fire conditions from network model input data.

performed (220 in this study and 1000 in the previous study). A Python script was developed to automatically generate the geometry within the ranges specified in the adjacent table. Room 2 geometry was constant with the room being 2.5 m wide, 2.5 m long, and 2.5 high. The openings on the east and west walls were set at 0.625 m wide by 1.9 m high. In Room 1, either the length or width was set to 2.5 m with the other dimension varied from 2.5 to 10 m. The openings in the three walls not connected to Room 2 were varied from 0.625 m wide by 1.9 m high, 1.25 m wide by 1.9 m high, or completely open. The fire could be located in either Room 1 or Room 2 and was varied in size from 50 to 1000 kW.

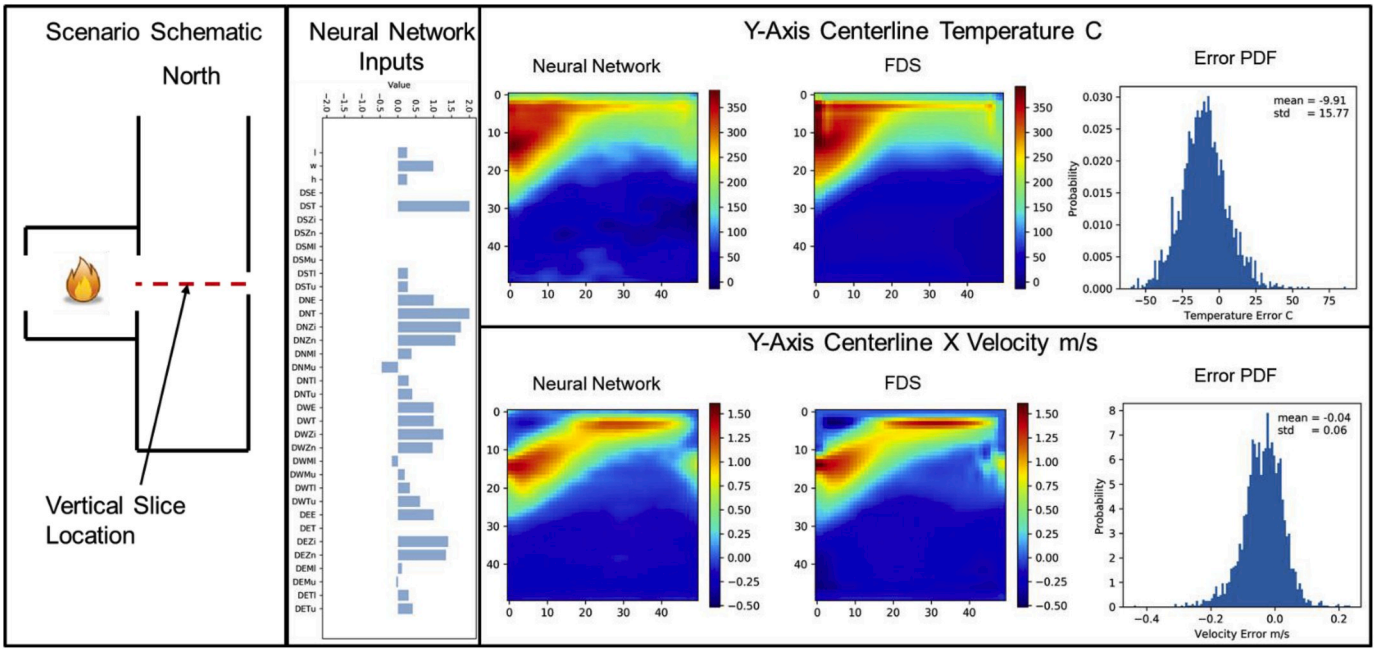


Fig. 9. Machine learning test results compared with FDS for centerline temperature and horizontal velocity at the slice location shown.

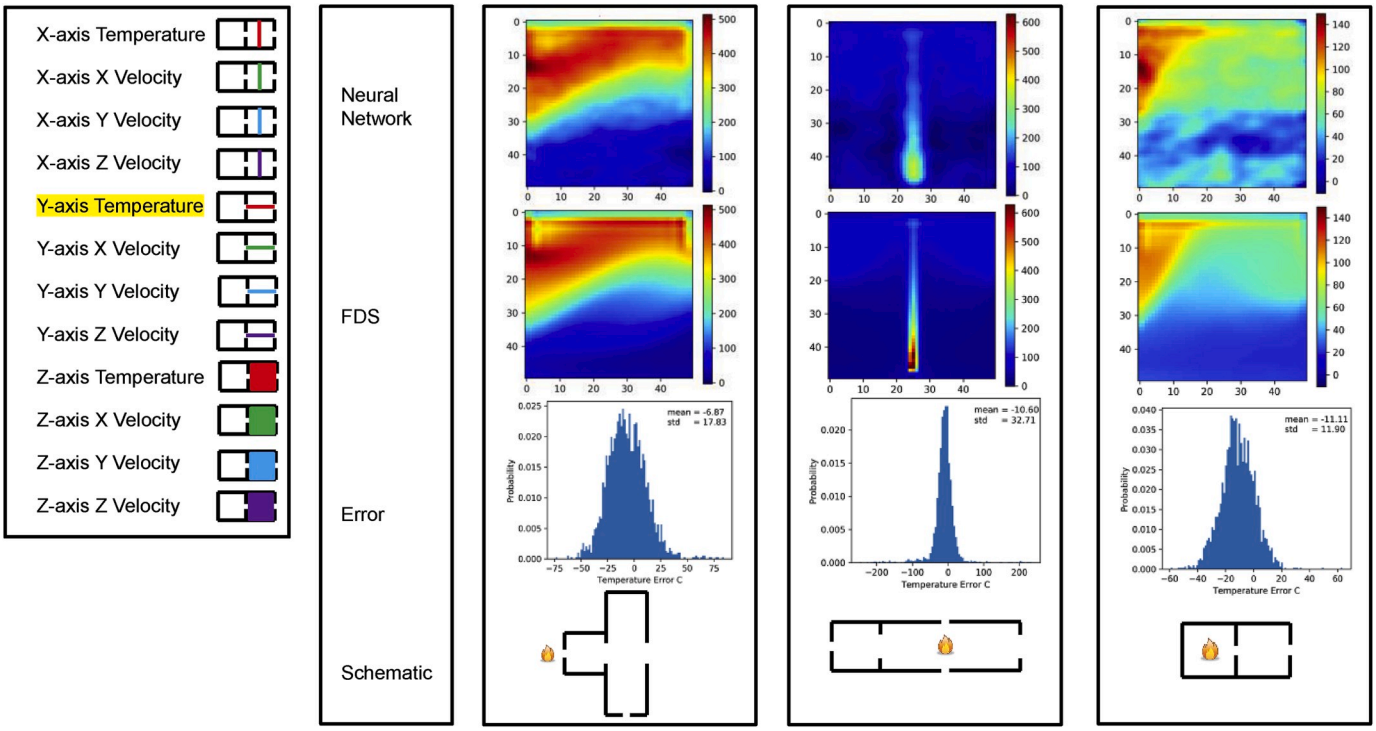


Fig. 10. Machine learning gas temperature results compared with FDS for different test data geometries.

The fire at Location 3 was meant to represent a fire located some distance away and outside of Room 2. In this case, the gas temperature, velocity, and interface height were selected to represent this condition.

All boundaries were taken as insulated, and FDS was used to predict 30 s of fire conditions. The FDS predicted conditions were used to calculate the volumetric averages within the rooms and the door flow rates that would be predicted using a network model. In addition, slice files were saved for a vertical distribution at the door connecting Rooms 1 and 2 (east-west), a vertical distribution along the length of Room 2 (north-south direction), and a horizontal distribution in Room 2 located 0.1 m below the ceiling. All quantities were average conditions determined by averaging the results from 10 to 30 s. The machine learning was performed using data (training and validation data sets) from 220 FDS simulations with 10 simulation results reserved as the test set.

The up-convolutional neural network architecture used in the machine learning is provided in Fig. 8. The model calculation proceeds from left to right. On the left side of the figure are the input values, which are the volumetric average quantities that would be obtained from a network model. The input is a vector pf 35 points. This includes three values corresponding to the Room 1 geometry (length, width, and height) and 32 values for each wall of Room 1 describing the details of the openings in the wall. This includes the opening size (0.625 × 1.9 m, 1.25 × 1.9 m, or full wall), ventilation state (open or closed), upper layer temperature, lower layer temperature, upper layer mass flow rate, lower layer mass flow rate, interface height, and neutral plane. These input parameters would come from a network model result. On the right-hand side are the outputs, which are the three velocity components and temperature for each of the three slice locations (total of 12 contour plots).

The eleven layers between the input and output are the multilayer of perceptrons that are used to learn the flux distributions using kernels (filters) based on the discrete input parameters. Note the "Drop Out" layer within this network, which is included to reduce the overfitting by the convolutional neural network. Prior to the output, a 2D Gaussian filter is provided to each distribution for smoothing. This was implemented using TensorFlow using the bindings from Python 3. Inputs and outputs were normalized before input to the TCNN to ensure the inputs and outputs were of a similar order of magnitude (temperatures divided by 1000, velocities and mass flow rates divided by 5, physical dimensions divided by 10). The weights and biases in the TCNN layers were initialized using the initializer presented by Golorot [46]. A fixed learning rate of 10^{-4} was used in this work.

A series of results from the machine learning up-convolutional neural network model (ANN) are provided in Figs. 9–17 based on test data (i.e., data not used to do the machine learning). Previous work for an ANN trained on 1000 simulation results showed two geometric cases and select temperature and velocity contours. These figures contain a wider range of geometric and fire parameter conditions as well as more output parameter comparisons. Each FDS simulation runtime was approximately 1800 s while the runtime for the ANN was <1 s, which is a computational speed up of 3 orders of magnitude. Overall, the ANN was determined to predict the gas temperatures within a standard deviation of ±8.5% while velocities were within ±13%.

The ANN was able to accurately capture the spatial distributions in the conditions when the fire scenario was varied. In Fig. 9, the ANN predicts the gas jet exiting the door and rising to the ceiling. In addition, the ANN predicts the velocity on the wall opposite the door where a recirculation flow exists. Results in Fig. 10 highlight the ability of the ANN to predict the flow field for other various geometries. In the first case to the left, the ANN provides an excellent predict of both the horizontal and vertical variations. In the middle case with a fire in Room 1 where the slice was taken, the ANN captures the overall distribution but under-predicts the gas temperatures in the fire. This result would indicate that more training data is required for this type of scenario to improve these results. In the case on the right, the ANN is able to generally predict the overall distributions but over-predicts the gas temperatures in the lower part of the space. This may be attributed in part due to the temperatures in this case being lower overall making the differences more noticeable. Nonetheless, these results would also indicate that more training data on lower temperature, smaller fire cases may be warranted.

Fig. 11 shows a schematic representation and detailed histogram of inputs to the neural network for 3 test cases. The detailed results provided in Figs. 12–17 provide a qualitative comparison of the ANN with

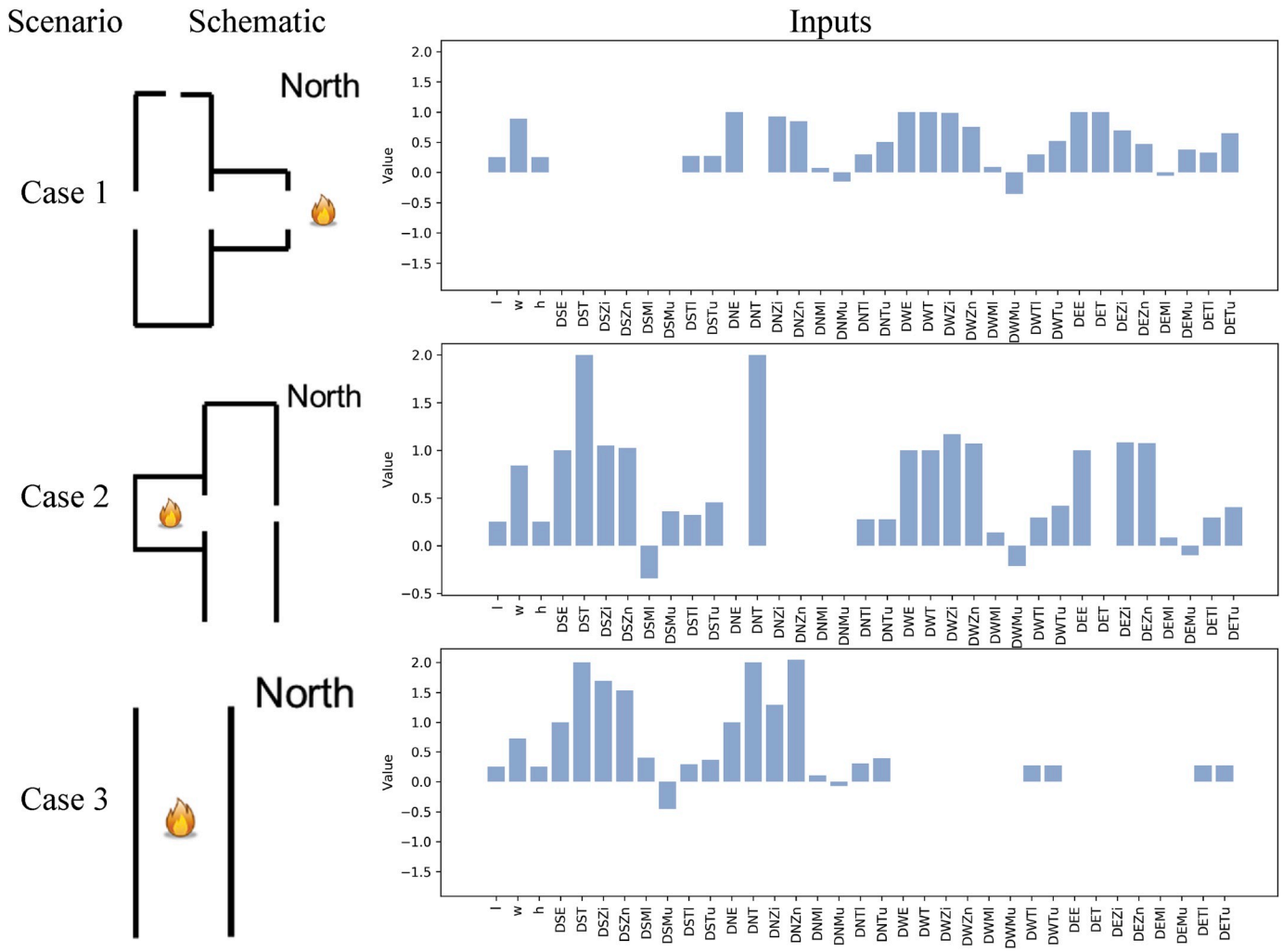


Fig. 11. Test cases schematic and input parameters.

results from FDS for all conditions and slices locations. As seen in these figures, overall the ANN is able to provide reasonable predictions for all the different test conditions at the different slice locations for different geometries. In some results where the velocities or temperatures are lower, the ANN predicts more variability than what is predicted using FDS. This is attributed to the limited training set size, which is being expanded in future work following this initial assessment.

Overall, the ANN appears to be able to successfully capture the spatial variations based on the discrete volumetric average quantities from the network model with minimal computational cost. Methods to expand this for use over larger domains with multiple rooms has been performed in related work with good comparison with CFD results [10]. For this case, it is seen through these results that the training data set can be reduced significantly (from 1000 in previous work to 220 in this work) and still provide an excellent representation of the flow field conditions. Methods are being explored to assess the minimal amount of input data to generate these predictions and whether the ANN architecture can be modified to reduce the number of input variables.

5. Conclusions

There is a need in fire applications to produce computationally efficient, high fidelity results to support a range of activities including hazard/risk assessments, understanding conditions based on sparse sensor data, and generating mitigation strategies for first responders. Machine learning is an approach that is being widely used to provide computationally efficient solutions for various types of problems. Research has shown promise in using dimensionality reduction techniques and deep learning to predict spatial distributions of fire conditions. Dimensionality reduction techniques require more mathematical development, which may hinder their more immediate use. Deep learning based on convolutional neural networks have been demonstrated to be highly capable of predicting spatial conditions. Basic architectures to do this based on discrete and full-field data inputs have been developed. With additional training and modification to these basic architectures, neural networks could be more rapidly expanded in their predictive capability and deployed for use. Increasing the amount of training data is a challenge which needs further investment; however, data generated from simplified geometry data can be used to predict more complex scenarios.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Brian Lattimer, who is now at Virginia Tech, has an ownership/equity interest in Jensen Hughes. This is stated in the acknowledgements of the paper.

CRediT authorship contribution statement

B.Y. Lattimer: Writing - review & editing, Writing - original draft. **J.**

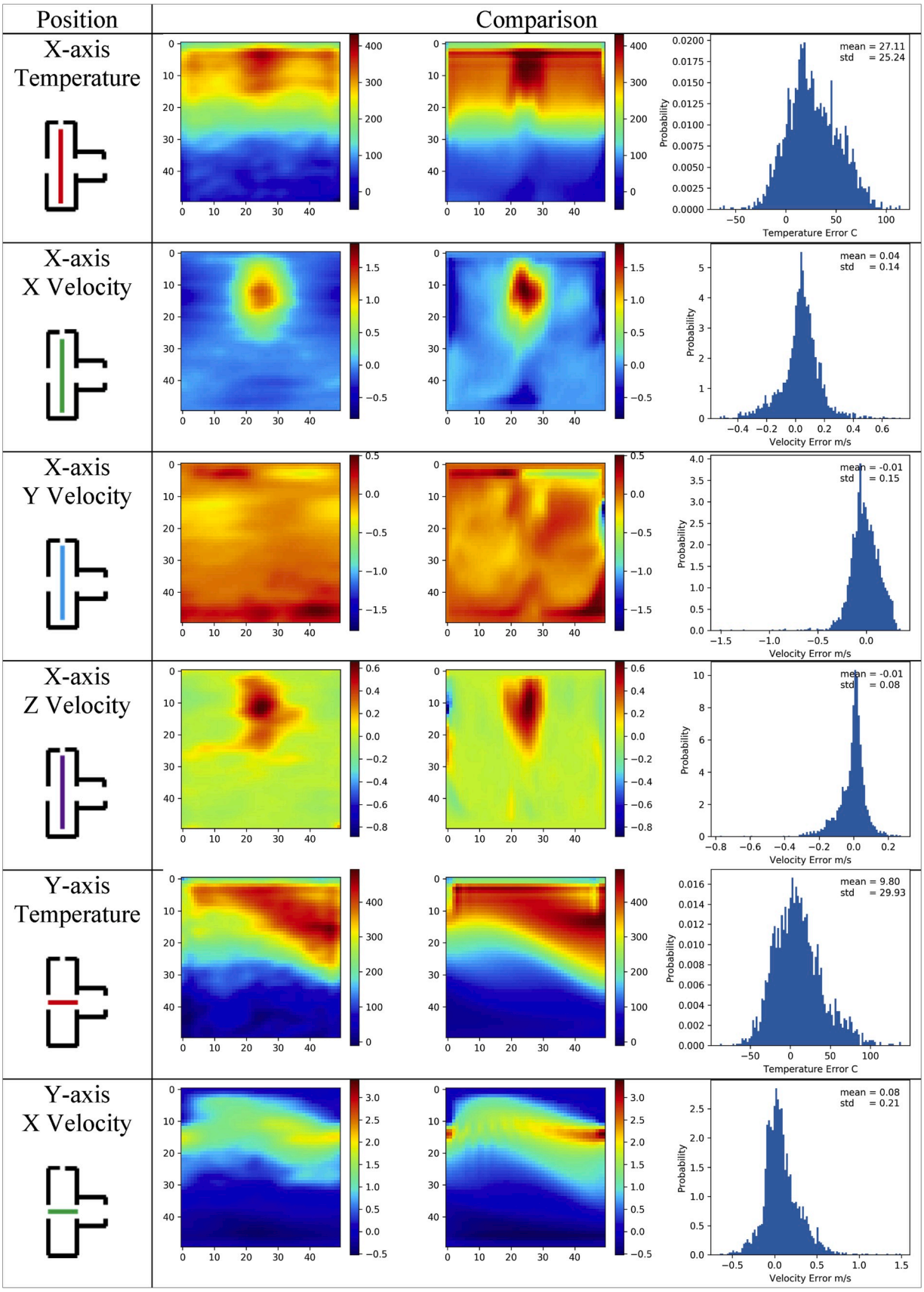


Fig. 12. Test Case 1 comparison part 1.

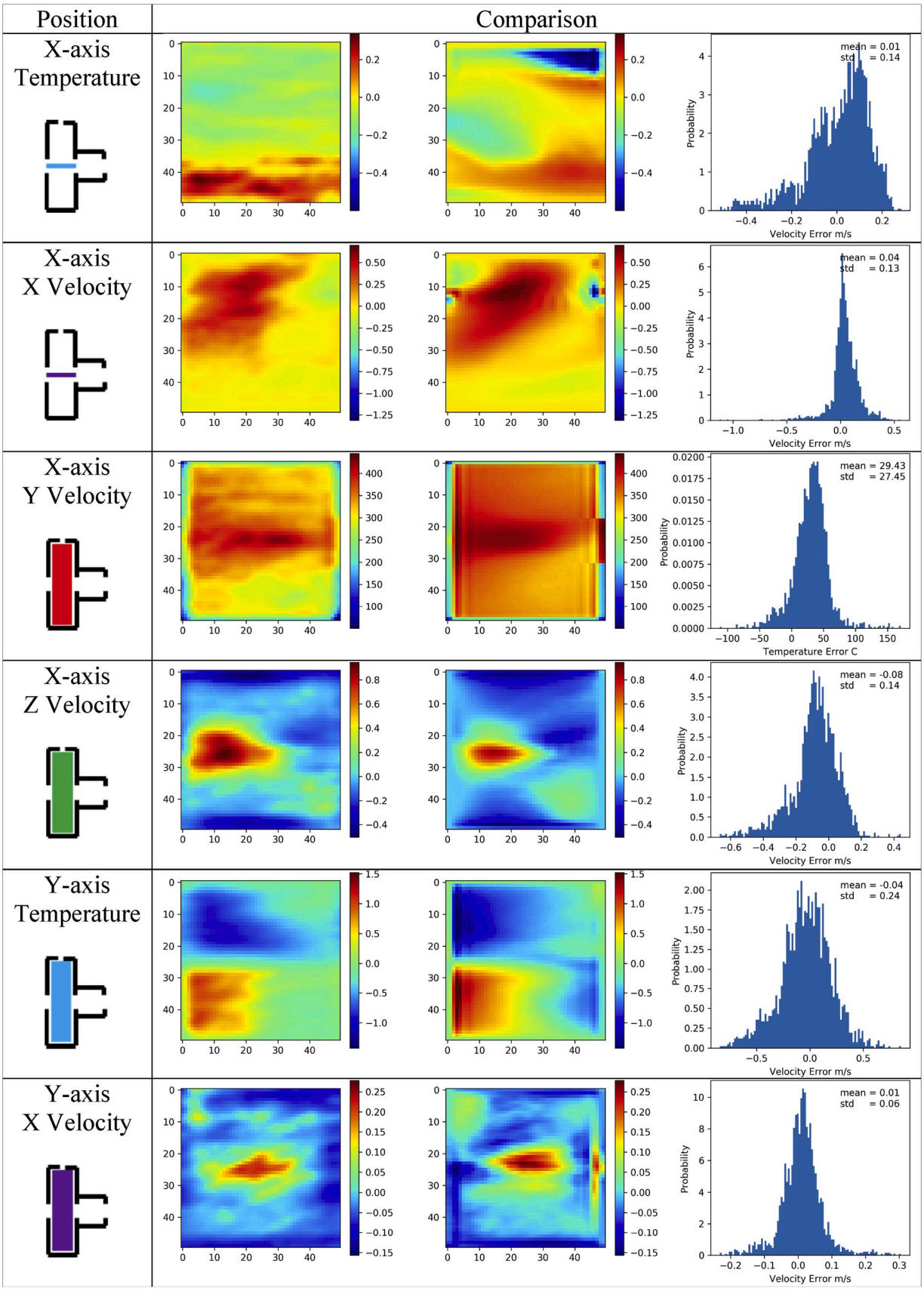


Fig. 13. Test Case 1 comparison part 2.

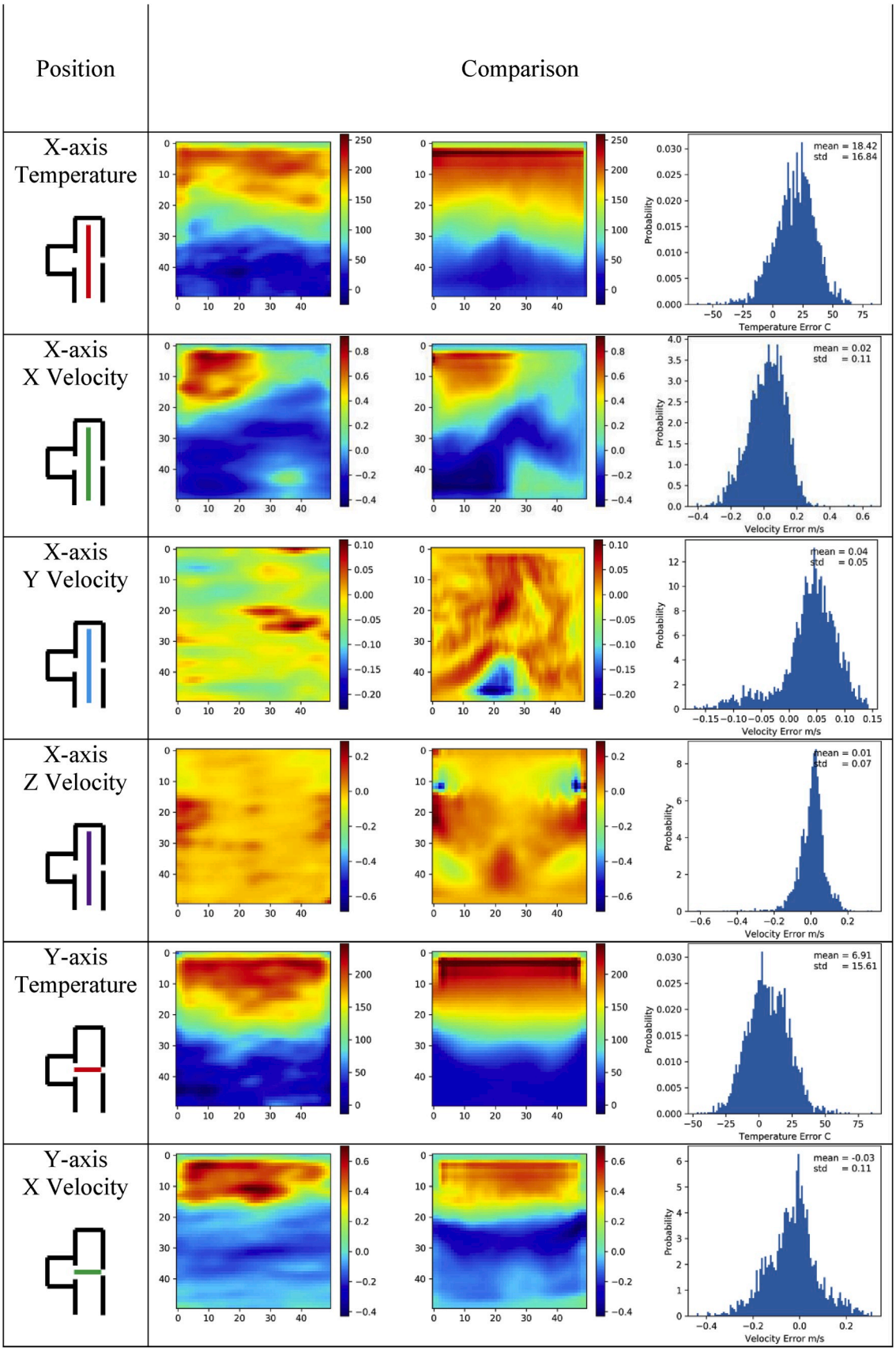


Fig. 14. Test Case 2 comparison part 1.

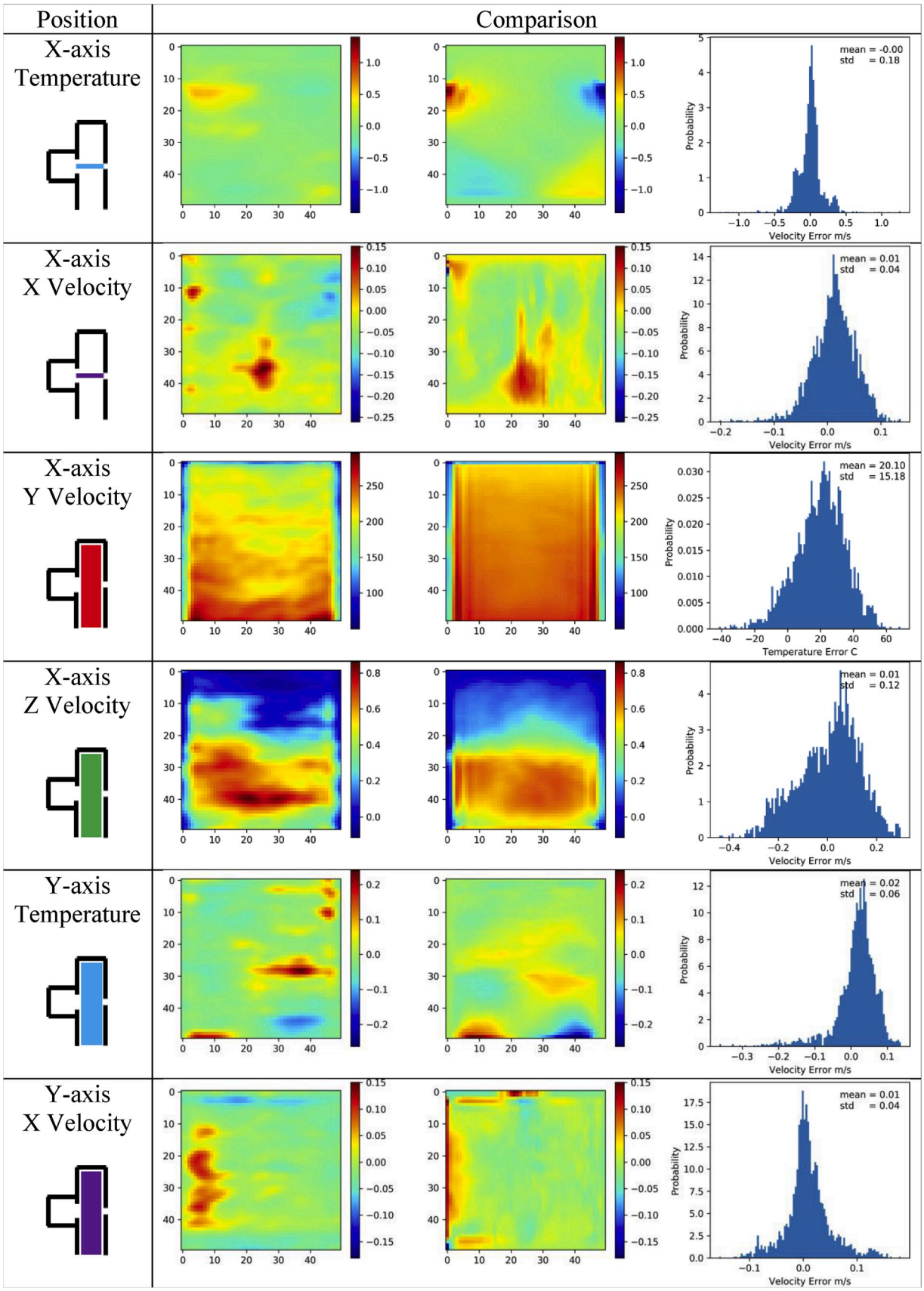


Fig. 15. Test Case 2 comparison part 2.

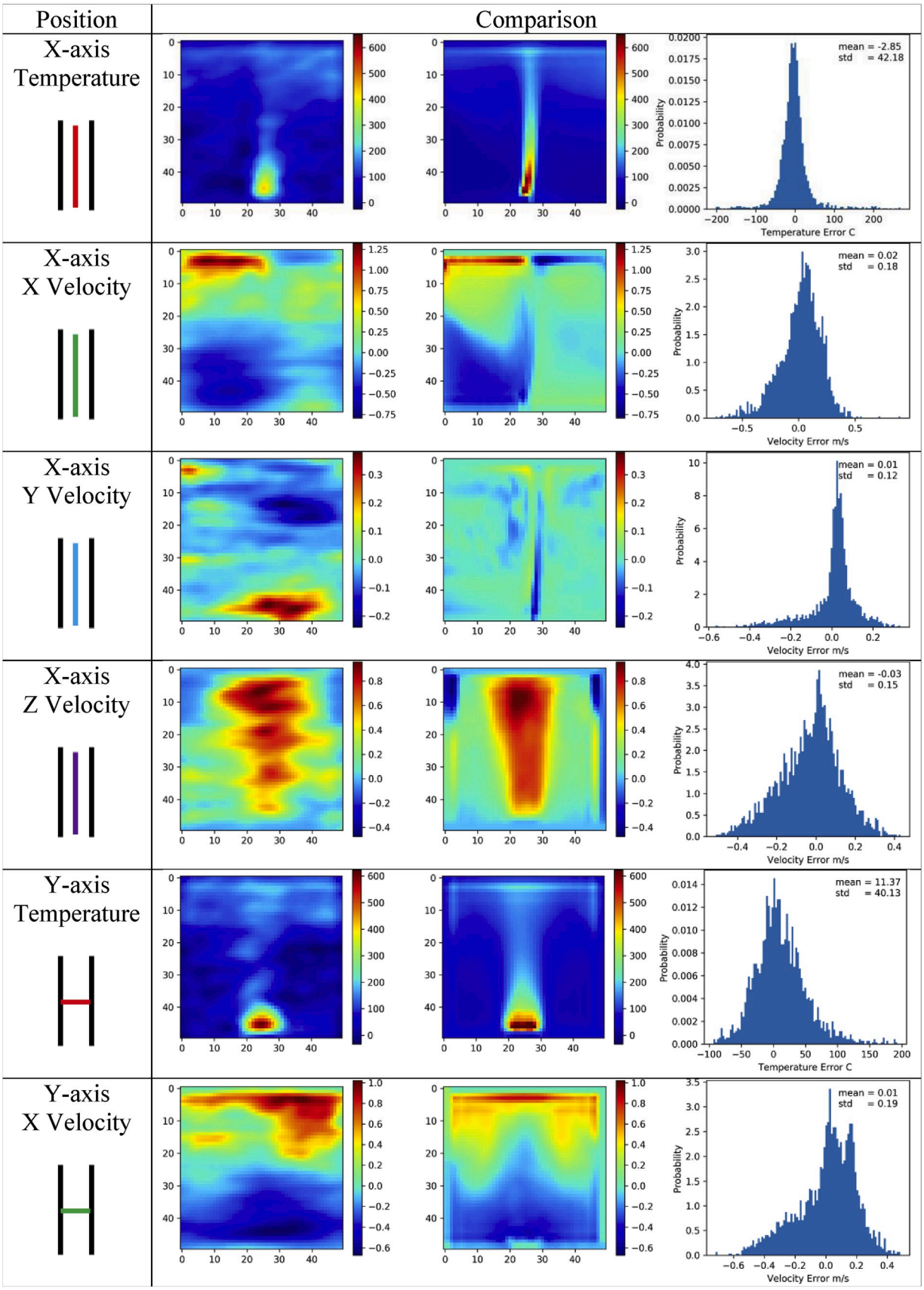


Fig. 16. Test Case 3 comparison part 1.

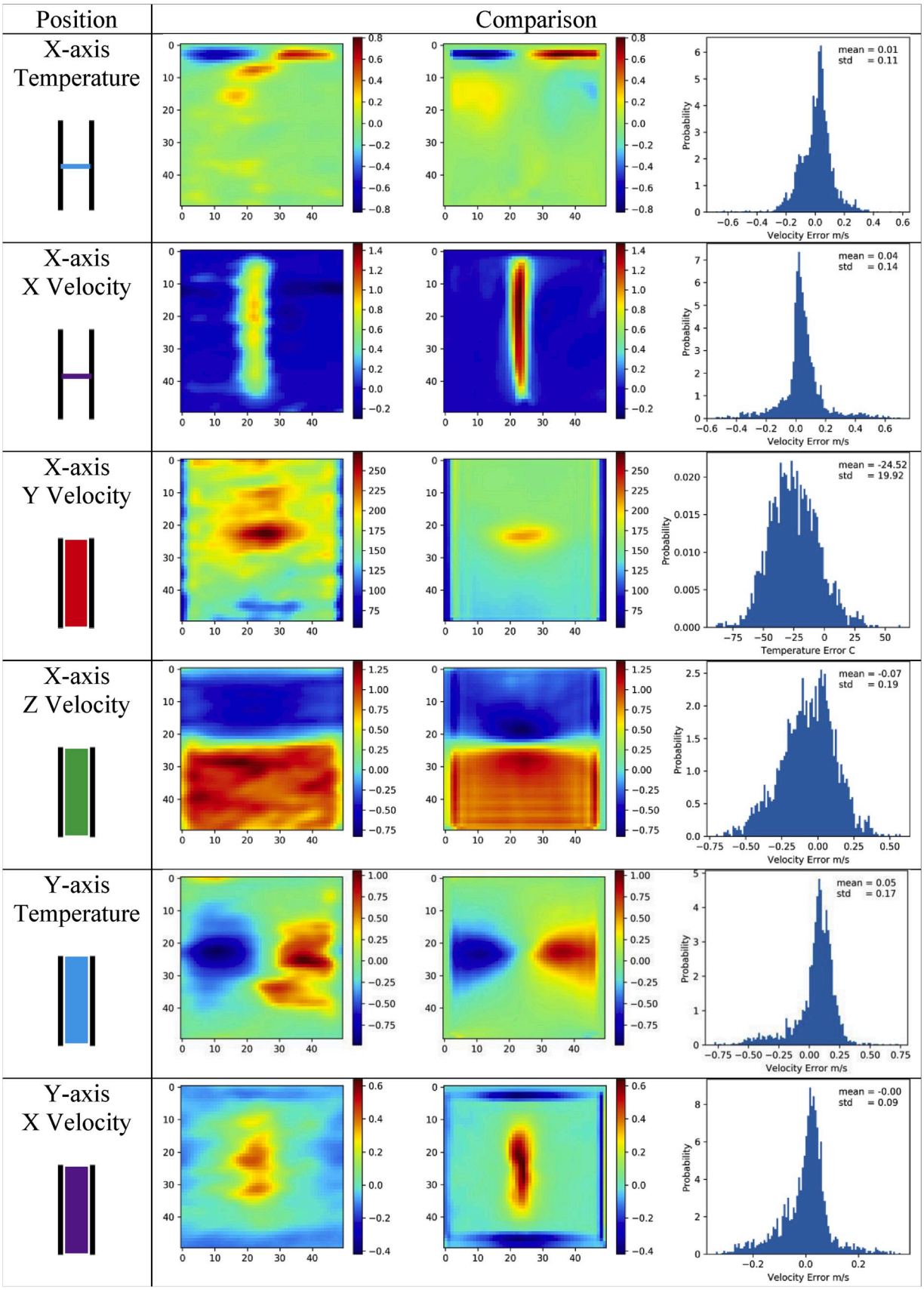


Fig. 17. Test Case 3 comparison part 2.

L. Hodges: Data curation.

Acknowledgements

This research was partially funded under NIOSH Grant No.200-2014-59669. The findings and conclusions in this paper are those of the authors and do not reflect the official policies of the Department of Health and Human Services; nor does mention of trade names, commercial practices, or organizations imply endorsement by the U.S. Government. The authors also appreciate the discussions and support of Prof. Kray Luxbacher in the Department of Mining&Minerals Engineering at Virginia Tech. In addition, the input and discussions from Prof. Jeffrey Borggaard and Prof. Serkan Gugercin on the dimensionality reduction portion of this effort is also appreciated. Brian Lattimer has an ownership/equity interest in Jensen Hughes.

References

[1] K. McGrattan, S. Hostikka, R. McDermott, J. Floyd, C. Weinschenk, K. Overholt, Fire Dynamics Simulator, User's Guide, sixth ed., vol. 1019, NIST Spec. Publ., 2013 https://doi.org/10.6028/NIST.SP.1019.
[2] Y. Wang, P. Chatterjee, J.L. de Ris, Large eddy simulation of fire plumes, Proc. Combust. Inst. 33 (2011) 2473–2480, https://doi.org/10.1016/j.proci.2010.07.031.
[3] R.D. Peacock, P.A. Reneke, G.P. Forney, NIST Technical Note 1889v2 CFAST – Consolidated Model of Fire Growth and Smoke Transport (Version 7) Volume 2 : User ' s Guide, vol. 2 ((n.d.)).
[4] J. Williamson, C. Beyler, J. Floyd, Validation of numerical simulations of compartment fires with forced or natural ventilation using the fire and Smoke simulator (FSSIM), CFAST and FDS, IAFSS 10 (2011) 1277–1288, https://doi.org/10.3801/IAF.
[5] F. Colella, G. Rein, V. Verda, R. Borchiellini, Multiscale modeling of transient flows from fire and ventilation in long tunnels, Comput. Fluids 51 (2011) 16–29, https://doi.org/10.1016/j.compfluid.2011.06.021.
[6] A. Haghighat, K. Luxbacher, B. Lattimer, Development of a methodology for interface boundary selection in the multiscale road tunnel fire, Fire Technol. 54 (2018) 1043–1080, https://doi.org/10.1007/s10694-018-0724-0.
[7] I. Vermesi, G. Rein, F. Colella, M. Valkvist, Reducing the Computational Requirements for Simulating Tunnel Fires by Combining Multiscale Modelling and Multiple Processor Calculation, (n.d.) 1–16..
[8] B. Ralph, R. Carvel, Coupled hybrid modelling in fire safety engineering ; a literature review, Fire Saf. J. 100 (2018) 157–170, https://doi.org/10.1016/j.firesaf.2018.08.008.
[9] E. Alpaydin, Introduction to Machine Learning, third ed., MIT Press, Cambridge, Mass, 2014, p. 593.
[10] J.L. Hodges, Predicting Large Domain Multi-Physics Fire Behavior Using Artificial Neural Networks, Virginia Polytechnic Institute and State University, 2018.
[11] J. Lumley, The structure of inhomogeneous turbulent flows, Atmos. Turbul. Radio Wave Propag. (1967) 166–178.
[12] A. Lattimer, Model reduction of nonlinear fire dyanmics models, PhD Diss. Virginia Tech. (2016).
[13] A. Lattimer, Model reduction of nonlinear fire dynamics models, PhD Diss. Virginia Tech, Math. (2016).
[14] A. Lattimer, J. Borggaard, S. Gugercin, K. Luxbacher, B. Lattimer, Computationally Efficient Wildland Fire Spread Models, Interflam (2016) 10, 2016.
[15] A. Lattimer, B. Lattimer, S. Gugercin, J. Borggaard, High fidelity reduced order models for wildland fires, 5th Int. Fire Behav. Fuels (2015) 8.
[16] Y. Safi, A. Bouroumi, Prediction of forest fires using artificial neural networks description of the proposed method, Artif. Neural Network. 7 (2013) 271–286.
[17] J. Storer, R. Green, PSO Trained Neural Networks for Predicting Forest Fire Size : A Comparison of Implementation and Performance, 2016, pp. 676–683.
[18] B.H. Naganathan, S.P. Seshasayee, J. Kim, W.K. Chong, J. Chou, Wildfire Predictions : determining reliable using fused dataset, Global J. Comput. Sci. Technol. 16 (2016).
[19] Y. Cao, M. Wang, K. Liu, Wildfire susceptibility assessment in southern China: a comparison of multiple methods, Int. J. Disaster Risk Sci. 8 (2017) 164–181, https://doi.org/10.1007/s13753-017-0129-6.
[20] R.J. McCormick, Toward a Theory of Meso-Scale Wildfire Modeling: A Complex Systems Approach Using Artificial Neural Networks, PhD Diss. Univ. Wisconsin-Madison., 2001, p. 169.
[21] G.P. Johnson, N.C.D.A. Station, F.C. Lin, Hurricane Tracking via Backpropagation Neural Network, ((n.d.)).
[22] T. Lee, Neural Network Predict. Storm Surge 33 (2006) 483–494, https://doi.org/10.1016/j.oceaneng.2005.04.012.
[23] C.M. Tseng, C.D. Jan, J.S. Wang, C.M. Wang, Appl. Artif. Neural Network. Typhoon Surge Forecast. 34 (2007) 1757–1768, https://doi.org/10.1016/j.oceaneng.2006.09.005.
[24] M.C. Deo, Artificial neural networks in coastal and ocean engineering, Indian J. Mar. Sci. 39 (2010) 589–596.
[25] M. Campolo, P. Andreussi, A. Soldati, River flood forecasting with a neural network model, Water Resour. Res. 35 (1999) 1191–1197, https://doi.org/10.1029/1998WR900086.
[26] C.W. Dawson, R. Wilby, Progress in physical geography, Prog. Phys. Geogr. 25 (2001) 80–108, https://doi.org/10.1177/030913330102500104.
[27] H.R. Maier, A. Jain, G.C. Dandy, K.P. Sudheer, Methods used for the development of neural networks for the prediction of water resource variables in river systems : current status and future directions, Environ. Model. Software 25 (2010) 891–909, https://doi.org/10.1016/j.envsoft.2010.02.003.
[28] S. Lek, J.F. Gue, Artificial neural networks as a tool in ecological modelling, an introduction 120 (1999) 65–73.
[29] H. Ashouri, K. Hsu, et al., Persiann-CDR: daily precipitation climate data record from multisatellite observations for hydrological and climate studies, Bull. Am. Meteorol. Soc. 96 (2015) 69–83, https://doi.org/10.1175/BAMS-D-13-00068.1.
[30] J.F. Mas, J.J. Flores, The Application of Artificial Neural Networks to the Analysis of Remotely Sensed Data, 2008, p. 1161, https://doi.org/10.1080/01431160701352154.
[31] Z.L. Langford, J. Kumar, F.M. Hoffman, Convolutional Neural Network Approach for Mapping Arctic Vegetation Using Multi-Sensor Remote Sensing Fusion, 2018, https://doi.org/10.1109/ICDMW.2017.48.
[32] S. Fan, J.R. Liao, R. Yokoyama, S. Member, L. Chen, S. Member, W. Lee, Forecasting the wind generation using a two-stage network based on meteorological information, IEEE Trans. Energy Convers. 24 (2009) 474–482.
[33] E. Izgi, A. Oztopal, B. Yerli, M. Kaymak, A. Sahin, Short – mid-term solar power prediction by using artificial neural networks, Sol. Energy 86 (2012) 725–733, https://doi.org/10.1016/j.solener.2011.11.013.
[34] T.P. Miyanawala, R.K. Jaiman, An Efficient Deep Learning Technique for the, ((n.d.)).
[35] S. Lee, D. You, Prediction of Laminar Vortex Shedding over a Cylinder Using Deep Learning, 2013.
[36] S. Lee, D. You, Data-driven prediction of unsteady flow over a circular cylinder using deep learning, ArXiv (2016) 1–35, 1804.06076v2.
[37] R. Maulik, O. San, A Neural Network Approach for the Blind Deconvolution of Turbulent Flows, 2017.
[38] J. Lee, S. Lee, D. You, Deep learning approach in multi-scale prediction of turbulent mixing-layer, ArXiv Prepr (2018). ArXiv 1809.07021.
[39] C. Chang, N. Dinh, Classification of machine learning frameworks for data-driven thermal fluid models, Int. J. Therm. Sci. 135 (2019) 559–579.
[40] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics Informed Deep Learning (Part I): Data-Driven Solutions of Nonlinear Partial Differential Equations, (n.d.) 1–22.
[41] M. Raissi, P. Perdikaris, G.E. Karniadakis, Physics Informed Deep Learning (Part II): Data-Driven Discovery of Nonlinear Partial Differential Equations, (n.d.) 1–19...
[42] S. Mishra, Computations of Differential Equations, 2018, pp. 1–23.
[43] J.L. Hodges, B.Y. Lattimer, K.D. Luxbacher, Compartment fire predictions using transpose convolutional neural networks, Fire Saf. J. 108 (2019) 102854, https://doi.org/10.1016/j.firesaf.2019.102854.
[44] A. Dosovitskiy, J. Springenberg, T. Brox, Learning to generate chairs with convolutional neural networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2017) 692–705.
[45] A. Dosovitskiy, J.T. Springenberg, M. Tatarchenko, T. Brox, Learning to generate chairs, tables and cars with convolutional networks, IEEE Trans. Pattern Anal. Mach. Intell. 39 (2017) 692–705, https://doi.org/10.1109/TPAMI.2016.2567384.
[46] X. Glorot, Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, J. Mach. Learn. Res. 9 (2010) 249–256.