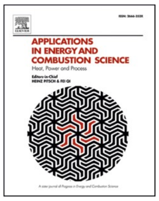
Applications in Energy and Combustion Science 16 (2023) 100211

Contents lists available at ScienceDirect

Applications in Energy and Combustion Science

journal homepage: www.sciencedirect.com/journal/applications-in-energy-and-combustion-science



Uncertainty quantification of a deep learning fuel property prediction model

Kiran K. Yalamanchi [a], Sahil Kommalapati [a], Pinaki Pal [a,*], Nursulu Kuzhagaliyeva [b], Abdullah S AlRamadan [c], Balaji Mohan [c,d], Yuanjiang Pei [e], S. Mani Sarathy [b], Emre Cenker [c], Jihad Badra [d]

[a] Argonne National Laboratory, Lemont, IL 60439, United States
[b] Clean Combustion Research Center (CCRC), Physical Science and Engineering Division, King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia
[c] Transport Technologies Division, R&DC, Saudi Aramco, Dhahran 31311, Saudi Arabia
[d] Aramco Research Center, R&DC, Saudi Aramco, Thuwal 23955, Saudi Arabia
[e] Aramco Americas: Aramco Research Center–Detroit, Novi, MI 48377, United States

A R T I C L E I N F O

Keywords:
Fuel property prediction
Deep learning
Uncertainty quantification
Monte Carlo ensemble methods
Bayesian neural network
Epistemic uncertainty
Aleatoric uncertainty

A B S T R A C T

Deep learning models are being widely used in the field of combustion. Given the black-box nature of typical neural network based models, uncertainty quantification (UQ) is critical to ensure the reliability of predictions as well as the training datasets, and for a principled quantification of noise and its various sources. Deep learning surrogate models for predicting properties of chemical compounds and mixtures have been recently shown to be promising for enabling data-driven fuel design and optimization, with the ultimate goal of improving efficiency and lowering emissions from combustion engines. In this study, UQ is performed for a multi-task deep learning model that simultaneously predicts the research octane number (RON), Motor Octane Number (MON), and Yield Sooting Index (YSI) of pure components and multicomponent blends. The deep learning model is comprised of three smaller networks: Extractor 1, Extractor 2, and Predictor, and a mixing operator. The molecular fingerprints of individual components are encoded via Extractor 1 and Extractor 2, the mixing operator generates fingerprints for mixtures/blends based on linear mixing operation, and the predictor maps the fingerprint to the target properties. Two different classes of UQ methods, Monte Carlo ensemble methods and Bayesian neural networks (BNNs), are employed for quantifying the epistemic uncertainty. Combinations of Bernoulli and Gaussian distributions with DropConnect and DropOut techniques are explored as ensemble methods. All the DropConnect, DropOut and Bayesian layers are applied to the predictor network. Aleatoric uncertainty is modeled by assuming that each data point has an independent uncertainty associated with it. The results of the UQ study are further analyzed to compare the performance of BNN and ensemble methods. Although this study is confined to UQ of fuel property prediction, the methodologies are applicable to other deep learning frameworks that are being widely used in the combustion community.

Introduction

The continuous development of fundamental databases and data acquisition from reacting flow experiments and simulations has, in recent years, been complemented with machine learning (ML) in the field of combustion [1]. With the growing amount of data available in the field, deep learning (DL), a subfield of machine learning that uses deep neural networks to train on large datasets, is being widely used. While it can be straightforward to quantify uncertainties in traditional physics-based models and simpler ML models to a certain extent, it is not the same case with complex models such as deep learning models. Quantifying the uncertainties by identifying sources of uncertainties in its predictions can make the predictions robust. Deep learning models with no uncertainty quantification (UQ) can lead to poor outcomes when these models are deployed in the decision-making process. This directly translates to lack of reliability of the deep learning model

* Corresponding author.
E-mail address: pal@anl.gov (P. Pal).

https://doi.org/10.1016/j.jaecs.2023.100211
Received 3 July 2023; Received in revised form 21 September 2023; Accepted 23 September 2023
Available online 24 September 2023

estimates. This can be undesirable and potentially hazardous when dealing with critical combustion applications. However, UQ for DL models incurs an additional computation cost. The added cost to the already computationally expensive DL models is one of the reasons that have hindered incorporation of UQ in most of the DL models developed in the combustion literature.

In this work, UQ of a deep learning-based fuel quantitative structure-property/activity relationship (QSPR/QSAR) model is performed. A QSPR model builds a relationship between the structural parameters of molecules and their chemical (and/or physical) properties. These models are used in several fields for predicting the properties of molecules and for designing new molecules. There have been several QSPR models developed over the years in the field of combustion. Benson's group additivity [2] is one of the simplest and widely used QSPR models for estimating thermodynamic properties using linear regression over the chemical groups present in the molecules. Over the years, the QSPR models moved from simple models such as linear and polynomial regression models to complex non-linear models such as k-nearest neighbors, support vector machine [3] and random forest regression [4]. This is driven by the effort to accurately fit the non-linearity in the data to better estimate the properties of the chemical molecules. However, the increasing dataset sizes and complexity in the data structures have made artificial neural networks a wider choice for developing QSPR models.

The neural network based QSPR models are developed to estimate several properties of fuel species, such as octane/cetane number [5–8], vapor pressure [9,10], heating values [11–13], flammability limits [14], and toxicity [15]. The input features to these models use either the molecular structure directly or descriptors generated from the structure. The improved accuracy with neural network models [16] along with their flexibility in handling several types of inputs such as multi-dimensional data for convolution neural networks and graph neural networks [17] has led to wider adaptation of neural networks for developing QSPR models. Recently, a fuel design workflow with a DL model to predict the properties of pure fuel components and mixtures of fuel components was developed by Kuzhagaliyeva et al. [18]. The DL model has shown improved accuracy over previous data-driven models, especially for fuel blends. The predictor network was combined with robust search algorithm to demonstrate inverse design of fuel formulations that yield high engine efficiency with lower emissions.

UQ has been used in the field of combustion for modeling uncertainties of chemical kinetic models [19,20] and reacting flow systems [21], among many other applications. There have been a lot of advances recently in UQ of deep learning models [22] and for DL based QSPR models [23–27]. However, the uncertainties in these QSPR models are often not investigated. This work aims to adapt the best principles from recent advancements in the field of deep learning to quantify uncertainties for fuel property prediction in the context of fuel design. By quantifying the degree of uncertainty in the predictions, it can be identified when the model is likely unreliable for a given fuel formulation, thereby supporting an informed decision making. Uncertainties from any estimation model can be broadly classified into two types. First is the uncertainty that is propagated from the underlying training data itself, which is known as aleatoric uncertainty. Second is the uncertainty induced by the model, which is known as epistemic or model uncertainty. Several combinations of model structure, hyperparameters and model parameters might be able to give a similar prediction accuracy; and this choices of parameters induces the epistemic uncertainty. In this work, quantification of both the aleatoric and epistemic uncertainties for DL models is discussed using multiple methods with different computational costs. It is noted that epistemic and aleatoric uncertainties are sometimes referred to as reducible and irreducible uncertainties respectively. However in this work, these terms are used in context of model and data uncertainties, respectively.

Bayesian neural network (BNN) is a type of deep learning model that provides uncertainty estimates for its predictions. BNNs offer UQ by means of a probabilistic interpretation of deep learning. In this probabilistic interpretation, the DL model weights are inferred as distributions rather than single values. Although BNNs are robust in providing UQ, training BNNs can be computationally expensive. The other approach to UQ for DL models is to use stochastic techniques, such as DropOut and DropConnect, that are widely used as regularization techniques. These regularization techniques combined with monte-Carlo sampling from different distributions offer a cheaper approach to UQ. It is also shown that these methods can theoretically be equivalent to BNNs; optimizing any neural network with these regularization techniques is equivalent to a form of approximate inference in a probabilistic interpretation of the model using BNN [28]. Both these approaches are employed in the current work and discussed in detail in subsequent sections.

The paper is organized as follows. First, the dataset used and the baseline DL model developed by Kuzhagaliyeva et al. [18] is discussed briefly. Next, the UQ models used in this study for quantifying epistemic and aleatoric uncertainties are described. Application of these UQ methods to the DL model is subsequently discussed. Thereafter, analysis of the results from these models with hyperparameter optimization is presented. Lastly, the main findings from the study are summarized in the Conclusions section.

Deep learning fuel property prediction model

A brief overview of the dataset and the baseline deep learning model used in this work for uncertainty quantification is discussed in this section; detailed description of the dataset and the DL model can be found in Kuzhagaliyeva et al. [18]. The database comprises single hydrocarbons as well as mixtures/blends, and was curated from a number of literature sources [29–33]. This consists of experimentally obtained measurements for three combustion-related properties, research octane number (RON), motor octane number (MON), and yield sooting index (YSI). Both RON and MON are standard measures of a fuel's resistance to knocking relative to n-heptane and isooctane reference fuels. Both these properties are experimentally measured in a cooperative fuel research (CFR) engine at operating conditions corresponding to ASTM standards [34,35]. YSI quantifies sooting propensity of a fuel based on the maximum soot volume fraction measured at the centerline of a coflow methane/air non-premixed flame doped with 400 ppm test fuel. This value is then converted to an apparatus-independent YSI using two selected reference compounds and scaling the volume fractions linearly within the assigned YSIs of reference compounds. In the curated database with 1159 data points, only 141 data points have all three measurements of RON, MON and YSI available, and the remaining 1018 observations have at least one missing property. Therefore, a customized stratified sampling was used to split the dataset to ensure that observations from all relevant subpopulations w.r.t. properties and pure components/mixtures were included in the training, validation, and test datasets.

The DL model consists of three smaller networks: Extractor 1, Extractor 2, and Predictor, and a mixing operator. The architecture of the DL model is shown in Fig. 1. Molecular fingerprints, that transform molecular structures into numerical strings, are first encoded through the use of Extractor 1 and Extractor 2. The Extractor 1 architecture has three stacked long short-term memory (LSTM) layers with descending dimensionality of output features (256, 64, and 16 neurons, respectively) and a fully connected layer with 12 neurons to transform the LSTM output. This Extractor 1 extracts a vector "SMILES fingerprint" from the one hot encoding of the SMILES (Simplified Molecular-Input Line-Entry System) representation of a species. Extractor 2 has three sequential fully connected (FC) layers with [1048, 512, 12] neurons in each layer sequentially and maps the Mordred descriptor of a species to a vector "Mordred fingerprint". Mordred descriptors were normalized using a min-max scaler before passing through the Extractor 2. In the next step, both the SMILES and Mordred fingerprints are concatenated to give one vector, referred to as latent space representation of pure

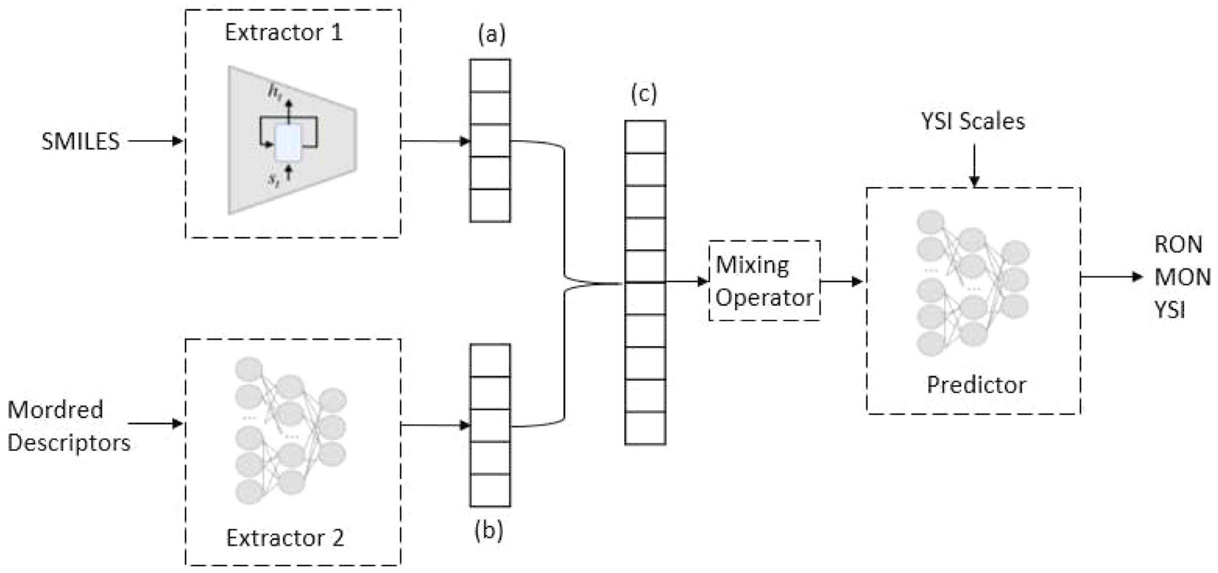


Fig. 1. Architecture of the baseline DL model [18] used in this work: (a) SMILES fingerprint and (b) Mordred fingerprint are concatenated to generate the (c) latent space representation of pure species. Latent space representation for a mixture is then obtained by weighted averaging of the latent space representations of indvidual species with corresponding volume fractions using the linear mixing operator.

components. In case of a mixture/blend, the Mixing Operator (MO) combines the latent space representations of pure components through linear weighted averaging. Subsequently, the predictor network, comprised of three FC layers with [128, 24, 12] neurons in each layer sequentially and with rectified linear unit (ReLU) activation function, and a final linear layer uses this combined latent space representation to predict the target properties of the blend. Since several numerical scales were used in the curated YSI database, their assigned YSI values are also provided as inputs to the predictor network.

Uncertainty quantification methods

Epistemic uncertainty: Bayesian neural networks

BNNs are a class of neural networks that incorporate Bayesian inference into the network architecture. Unlike traditional neural networks that produce point estimates for the model parameters, BNNs represent the parameters as probability distributions, allowing them to capture uncertainty in their predictions. This means that rather than simply outputting a single prediction for a given input, a BNN can provide a distribution of possible predictions and their associated probabilities [36]. BNNs are composed of Bayesian layers that incorporate Bayesian inference by specifying a prior distribution over the layers' weights and biases, and then estimating a posterior distribution over these parameters using variational inference [37]. In the context of the present study, while an architecture full of Bayesian layers for all of Extractor 1, Extractor 2, and Predictor networks can be better at modeling the uncertainties, this can be complex and computationally expensive.

Non-Bayesian layers typically use a fixed set of weights and biases that are learned through standard backpropagation during training. These layers are computationally inexpensive to train relative to that of Bayesian counterparts and are commonly used in deep learning applications. In a hybrid network, Bayesian layers can be placed at the end of the network thereby using variational inference only for the weights of the last layers, along with non-Bayesian layers for other layers [38]. The combination of Bayesian and non-Bayesian layers can allow a neural network to take advantage of the strengths of both types of layers [39]. This can provide uncertainty estimates, while still maintaining computational efficiency during training and inference. Therefore, in this study, Bayesian layers are added to the predictor network only, while keeping the layers of Extractor 1 and Extractor 2 networks non-Bayesian.

Epistemic uncertainty: stochastic techniques

Stochastic techniques that are widely used for regularization in ANNs have recently emerged as powerful tools for quantifying uncertainty in deep learning models. These techniques are based on generating random samples for the posterior distribution over the model's weights and biases, which can be used to estimate model uncertainty. Stochastic techniques are increasingly being used in the UQ of deep learning models for a range of applications, providing model uncertainties without a large increment in computational cost. In this work, two stochastic techniques, DropOut and its variant DropConnect, are explored for UQ in the deep learning model. DropOut involves randomly dropping out a fraction of the neurons in a network during training. Post training, DropOut generates multiple models with different architectures for inference. These models generate multiple predictions, the combination of which are then used to estimate model prediction and its uncertainty during inference. Similarly, DropConnect involves randomly dropping out connections between neurons in a network during training and generating multiple models for uncertainty estimation post training. The distribution from which the neurons or connections are dropped out (using Monte Carlo technique) determines the method. In this work, Monte Carlo- Bernoulli DropOut, Bernoulli DropConnect, Gaussian DropOut and Gaussian DropConnect methods are explored. Fig. 2 depicts these stochastic methods. Gal et al. [28] showed that these stochastic techniques are theoretically equivalent to BNNs with a specific choice of approximating distributions for variational inference; for additional details of this proof, readers are directed to Gal et al. [28].

Aleatoric uncertainty

Both the aforementioned Bayesian neural networks and stochastic techniques can estimate only the uncertainties induced by the deep learning model. This does not include the uncertainty that might be induced from the underlying training dataset. The type of uncertainty that arises from the inherent randomness in the data itself is referred to

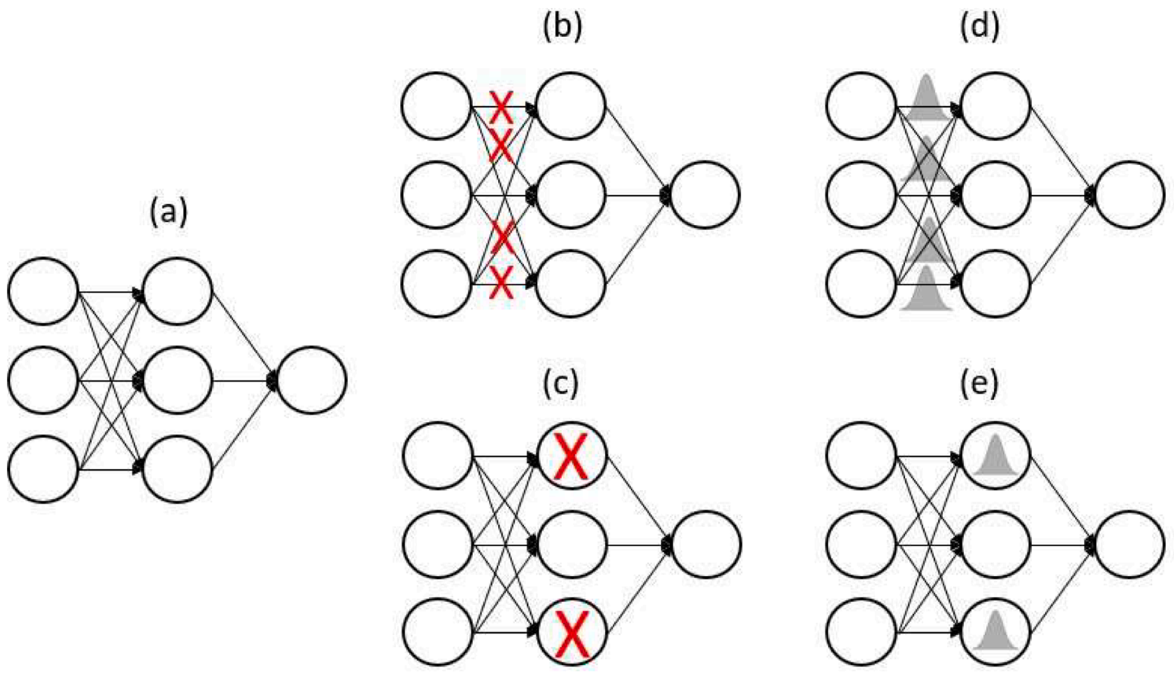


Fig. 2. Stochastic techniques used in this work for UQ – (a) Baseline model (b) Bernoulli DropConnect (c) Bernoulli DropOut (d) Gaussian DropConnect and (e) Gaussian DropOut.

as Aleatoric uncertainty. Modeling aleatoric uncertainty is particularly useful if the data has high variability. In the context of deep learning, aleatoric uncertainty can be modeled by modifying the loss function to include the data uncertainty or optimize the uncertainties of each data point independently. In this work, since there are no prior uncertainties for the data available, we will adopt the latter to model aleatoric uncertainty. For this, it is assumed that for each property and data point, the uncertainty value corresponds to the value that is obtained from treating it as Gaussian random variable. The aleatoric uncertainty modeling can be further divided into two types. Homoscedastic uncertainty modeling assumes that uncertainty in the three properties RON, MON, and YSI, does not vary within each property and only differs between properties. Heteroscedastic uncertainty modeling, on the other hand, assumes that each property prediction for a data point has an independent uncertainty associated with it.

$$L(W,\sigma_1,\sigma_2,\sigma_3) = \frac{1}{2\sigma_1^2}L_1(W) + \frac{1}{2\sigma_2^2}L_2(W) + \frac{1}{2\sigma_3^2}L_3(W) + log\sigma_1\sigma_2\sigma_3 \quad (1)$$

where $\mathscr{L}_i(W)$ is $\sum_{j=1}^{N} \| y_j - \mathrm{f}^w(\mathrm{x}_j)\|^2$ for property *i*, *N* is the number of data points for that property and σ is the standard deviation of each property prediction.

$$\mathscr{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{2\sigma(x_i)^2} \| y_i - \mathrm{f}(\mathrm{x}_i) \|^2 + log\sigma(x_i) \quad (2)$$

Where *i* is a data point and $\sigma(x_i)$ is corresponding standard deviation for that data point.

Eq. (1) corresponds to the loss function for modeling homoscedastic uncertainty with the three properties, viz RON, MON, and YSI. *y* is the experimental value of a property for a data point and $\mathrm{f}^w(x)$ is the predicted value from the model. The uncertainty for each of the three properties $(\sigma_1, \sigma_2, \sigma_3)$ in the equation are independently varied to optimize the loss function. Eq. (2) shows the loss function for modeling heteroscedastic uncertainty. In Eq. (2), uncertainty corresponding to each data point ($\sigma(x_i)$ for data point *i*) is an additional output to the model and is learned by minimizing the loss function. In both the cases, the loss function is modified from the mean squared loss function by scaling each mean squared difference between prediction and target by the corresponding uncertainty. Furthermore, a regularization term on uncertainties to contain their values to lowest possible values is added. This is also known as the Gaussian negative log likelihood (GNLL) loss that reflects the deviation of targets from their means and variances, as per the stated assumption of treating the targets as Gaussian random variables. In this work, only heteroscedastic aleatoric uncertainty is considered. Incorporation of homoscedastic aleatoric uncertainty within the UQ framework will be part of future work.

Results and discussion

Stochastic techniques over optimized network

Stochastic techniques can be readily applied over any optimized deterministic deep learning model network. However, optimization of the drop probabilities and re-optimization of the deep learning model architecture for optimal hyperparameters is required when stochastic techniques are applied for UQ of DL models. The optimal architecture could be different when the stochastic methods are applied to DL models and this warrants re-optimization of architecture. The optimization study and corresponding results are discussed in the next section. In this section, stochastic techniques are applied over the deep learning model without any hyperparameter optimization, keeping the model architecture same as that of baseline model described in Section 2. The dropout/drop-connection probabilities of the first layer of predictor network in the original model are set to 0.01 and the results are generated from training with different stochastic methods. The training, validation and test datasets are same as the ones used for the original model and the results are discussed for the test set. Table 1 shows the comparison of results from four different methods discussed in previous sections, viz., Monte Carlo- Bernoulli DropOut, Bernoulli DropConnect, Gaussian DropOut and Gaussian DropConnect. In particular, Table 1 provides the mean absolute error (MAE), confidence intervals (1-, 2- and 3- σ) and gaussian negative log likelihood loss for the three properties RON, MON and YSI corresponding to the four stochastic methods considered. The confidence intervals in the Table 1 indicate the percentage of test data values that are within the 1-, 2- and 3- σ of the test set predictions. For the fixed drop probabilities, it is observed that the

Table 1
Comparison of the results from four different stochastic methods without hyperparameter optimization (MAE – Mean Absolute Error (the values in parentheses indicate MAE for each of the three properties), CI – Confidence Interval (the three values indicate the percentage of data in 1-, 2-, 3- σ CI, respectively), GNLL – Gaussian Negative Log Likelihood (the values in parentheses indicate GNLL for each of the three properties)).

Model Description	MAE (RON/MON/YSI)	RON CI (1, 2, 3- σ)	MON CI (1-, 2-, 3- σ)	YSI CI (1-, 2-, 3- σ)	GNLL (RON/MON/YSI)
MC Bernoulli DropOut	13.34 (3.66, 3.29, 6.3)	27%, 45%, 59%	26%, 45%, 55%	8%, 13%, 16%	2113 (17, 20, 2075)
MC Bernoulli DropConnect	14.22 (3.1, 3.0, 8.01)	25%, 45%, 56%	22%, 43%, 55%	4%, 13%, 16%	741 (23, 26, 691)
MC Gaussian DropOut	13.24 (4.02, 3.18, 6.02)	26%, 50%, 62%	32%, 51%, 67%	2.7%, 10%, 16%	394 (19, 15, 358)
MC Gaussian DropConnect	12.06 (3.37, 3.20, 5.4)	29%, 53%, 68%	28%, 48%, 67%	9%, 19%, 26%	167 (11, 16, 140)

GNLL is lower for DropConnect compared to DropOut. Among the four methods, Gaussian DropConnect shows better performance in most of the metrics. Note that this however may not be reflective of the performance of each of the methods since the probabilities and architecture are not optimized. However, due to the computational cost, only the Gaussian DropConnect method is selected for the final UQ studies with hyperparameter optimization.

Hyperparameter optimization

The optimal architecture of a neural network can change when the stochastic and Bayesian methods are applied to a deep learning model. In this section, hyperparameter optimization of the modified DL models for uncertainty quantification is discussed. As mentioned in the previous section, only the predictor network of the original model is modified and both the extractor networks are kept the same. However, note that during training for optimal hyperparameters of the predictor network, all the weights including those of both the extractor networks are retrained. In view of the computational cost, hyperparameter optimization is conducted only for Gaussian DropConnect model among the stochastic models. Moreover, hyperparameter optimization is also done for the BNN model.

Table 2 shows the ranges of the hyperparameters varied for each of the two models. Note that for the BNN model, the probability range only corresponds to the probability used in the last layer of predictor network as the Bayesian layer is only applied for the last layer. In addition, the loss functions of both the Gaussian DropConnect and BNN are modified to include heteroscedastic aleatoric uncertainty.

The hyperparameter optimization is performed using *k*-fold cross validation (k-fold CV). *k*-fold CV is performed by splitting the entire training data (which is the entire dataset minus the test set) in *k* sets and using each one of them as the validation set while the remaining are used for training the model, in a loop until all the *k* sets are used as validation set once. The validation errors over all the *k* sets are averaged and used as the metric to choose the best hyperparameter combination. In this study, *k* is set to be 5 based on an ablation study, after considering the computational cost of training models for higher *k* values. For the hyperparameter search, an open source Bayesian optimization package, bayes_opt package [40], is used. This method starts with building initial runs over certain random hyperparameter combinations within the considered hyperparameter space. After the initial trials, at each subsequent trial, the gaussian process is fitted and posterior distribution along with exploration strategy is used to determine which combination of parameters to try next, until the total number of iterations reaches a user-specified limit.

The optimal architectures and probabilities of the Gaussian DropConnect and BNN models found from hyperparameter search are listed in Table 3. Both the models show optimal performance with three layers and decreasing number of neurons in the successive layers. Additionally, for Gaussian DropConnect model, the optimal probabilities increase with the successive layers. The optimal hyperparameter combinations are subsequently used to train a model with the entire training data and produce the outputs on test set. The test results are discussed in detail in the next section.

Table 2
Hyperparameter ranges used for hyperparameter optimization (HPO) of the predictor network.

Model	Number of layers	Neurons in each layer	Probabilities
Gaussian DropConnect + Aleatoric	1–6	16–2000	0–1
Bayesian Neural Network + Aleatoric	1–6	16–1500 (first layer) 16–512 (other layers)	10^{-4}–0.1

Model performance

The Gaussian DropConnect and BNN models with modified loss functions to include heteroscedastic aleatoric uncertainty are trained with the optimal hyperparameters obtained from 5-fold CV. The predictions with these two models for the test set provides the predictions with their uncertainties. Fig. 3 shows the comparison between the actual values and predictions from the Gaussian DropConnect model along with standard deviations of the predictions corresponding to the test set for RON, MON and YSI. In general, it can be observed that the predicted uncertainty is higher in regions of the property space where data is sparse. Although the true values seem to be within the predicted uncertainty, exceptions can also be observed mainly for RON and MON. Fig. 4 shows the comparison between the actual values and predictions from the BNN model along with the prediction standard deviations for the test set. The same observations as above are also hold true for BNN model. Fig. 5 shows variation of standard deviation with respect to the property's actual values for all the three properties and for both the models. Also shown in Fig. 5 are the plots of data density with respect to the property values. The inverse relationship of the data density and uncertainty is more evident in Fig. 5, for all the three properties. It can be observed from Fig. 5 that the major difference between both the models is that the BNN model tends to estimate lower uncertainty compared to Gaussian DropConnect model in regions of the property space where the data is clustered, for RON and MON. Furthermore, a separate analysis of the UQ results indicates that the aromatic species have higher estimated uncertainties relative to the other classes of compounds in the dataset.

Table 4 provides a comprehensive comparison of the performance of Gaussian DropConnect and BNN models for the test set. The mean absolute errors for both the models are similar and are also close to that of the original model from Kuzhagaliyeva et al. [18], which is 11.74 (3.05

Table 3
The optimal architectures obtained from hyperparameter optimization of the DL models used in this study.

Model	Number of layers	Neurons in each layer	Probabilities
Gaussian DropConnect + Aleatoric	3	1310/457/140	0.005/0.05/0.2
Bayesian Neural Network + Aleatoric	3	1214/213/98	0.001

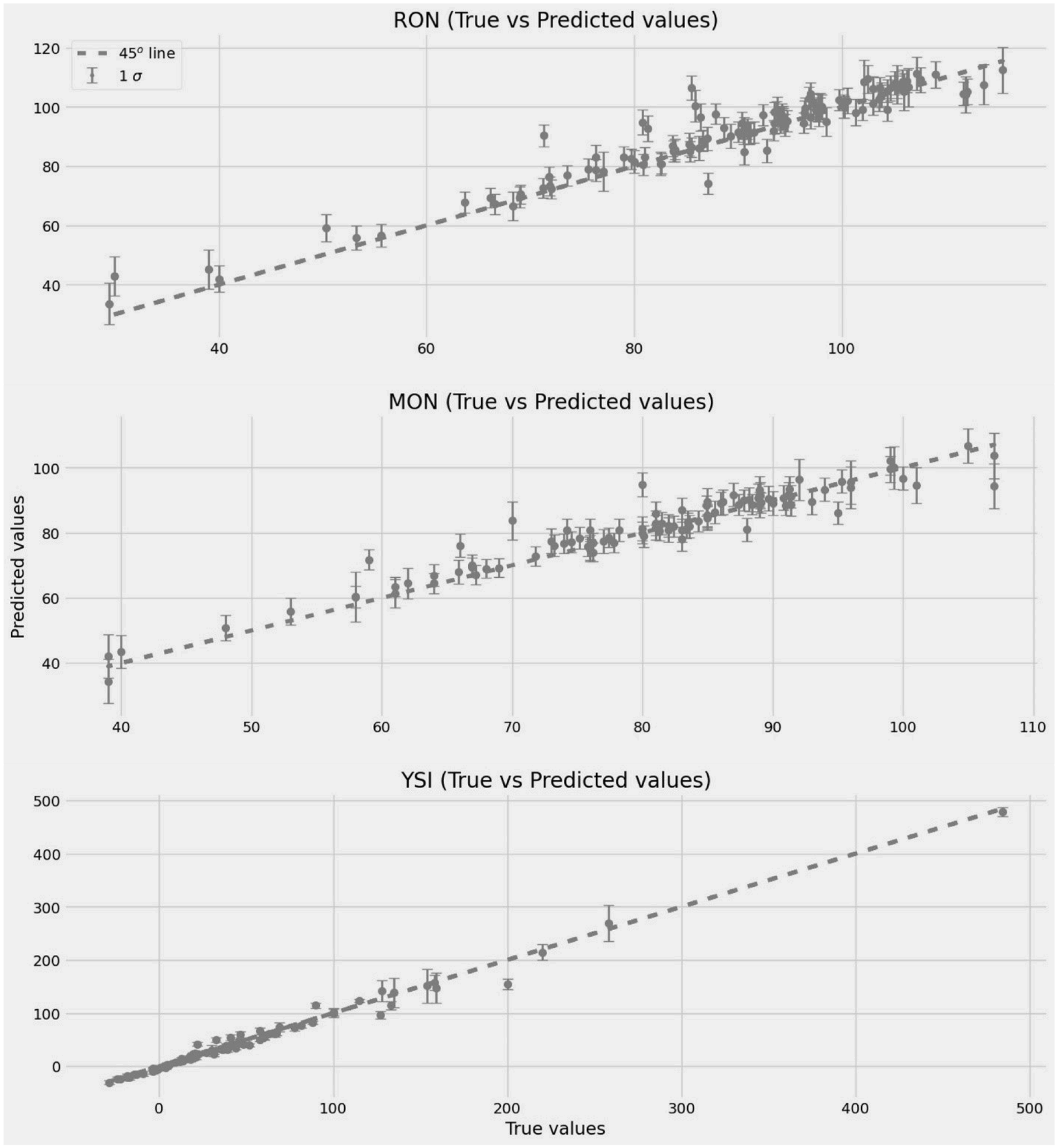


Fig. 3. Comparison of predicted values with true values for the three properties with their predicted uncertainties from Gaussian DropConnect model.

for RON, 2.79 for MON, 5.89 for YSI). This shows that the GNLL loss is optimized without significant impact on mean absolute error. In comparison with the Table 1 with the non-optimized architecture and drop probabilities, the GNLL for the optimal Gaussian DropConnect model is significantly improved. Also shown in Table 4 are the confidence intervals for all the three property predictions from both the models. The lower confidence intervals for RON and MON in case of BNN compared to Gaussian DropConnect model can be linked to the prior observation that the BNN model tends to estimate lower uncertainty relative to Gaussian DropConnect model in regions of the property space where the data is clustered. Nevertheless, this does not have significant impact on GNLL as GNLL of both the models is similar for both RON and MON. On a different note, it is also found (not shown in Table 4) that for both models, the epistemic uncertainty component is much smaller than the aleatoric component in the final predicted uncertainties, with the difference between the two components being more pronounced for Gaussian DropConnect.

Training for both BNN and Gaussian DropConnect were performed on a single NVIDIA A100 GPU. The computational cost of UQ with Gaussian DropConnect was similar to that of the baseline model since there was no increment in the trainable parameters during training. The marginal increment in cost during inference was insignificant relative to training time. Approximately 30 minutes of training time was required for each run of hyperparameter combination for Gaussian DropConnect. On the other hand, the BNN takes approximately twice as many epochs as the Gaussian DropConnect model to converge during training. Therefore, this doubled the computational cost of hyperparameter optimization for BNN relative to Gaussian DropConnect. However, since the hyperparameter space was limited for BNN to small hyperparameter ranges with only the last layer considered to be Bayesian, it took

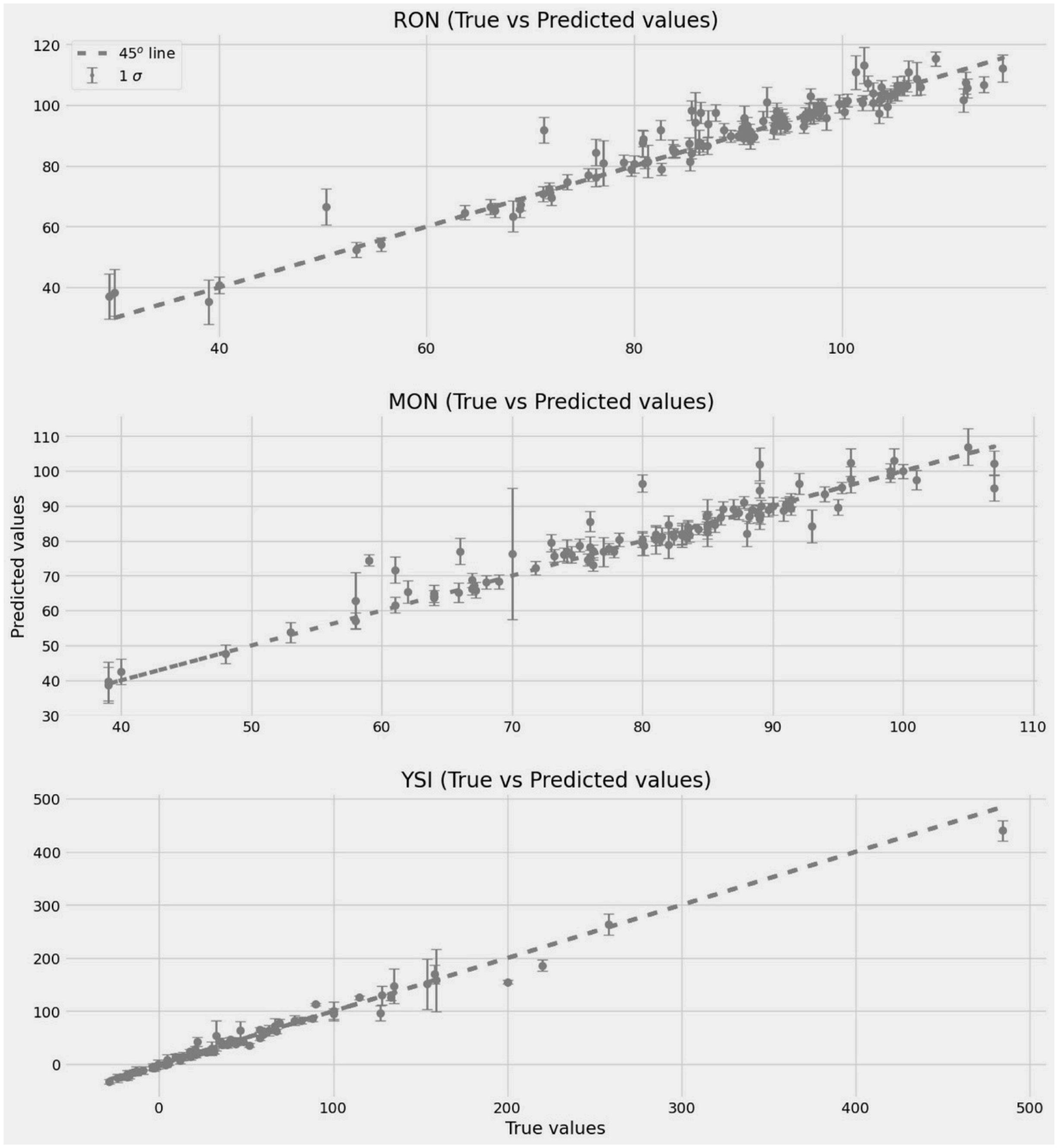


Fig. 4. Comparison of predicted values with true values for the three properties with their predicted uncertainties from BNN model.

approximately 45 min for each run of hyperparameter combination.

Conclusions

In the present work, UQ of a deep learning fuel property prediction model is performed. Gaussian DropConnect and BNN are used for modeling epistemic uncertainty. For both these models, the loss functions are further modified to take into account the uncertainty that arises from the data itself, known as aleatoric uncertainty. Hyperparameter optimization is conducted for both the models with 5-fold CV. The final optimized models are tested on the same test set that was used by the baseline fuel property prediction model. The following is a summary of the major outcomes from this work.

- The GNLL loss for the test set is quite reasonable and similar, without significant deterioration of mean absolute error, for both Gaussian DropConnect and BNN models.
- The computational cost is higher for BNN relative to Gaussian DropConnect with no significant advantage w.r.t. accuracy, making Gaussian DropConnect a better option for UQ.
- The quantified uncertainties can be used downstream in the fuel design process to make informed choice for exploration vs exploitation.

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.

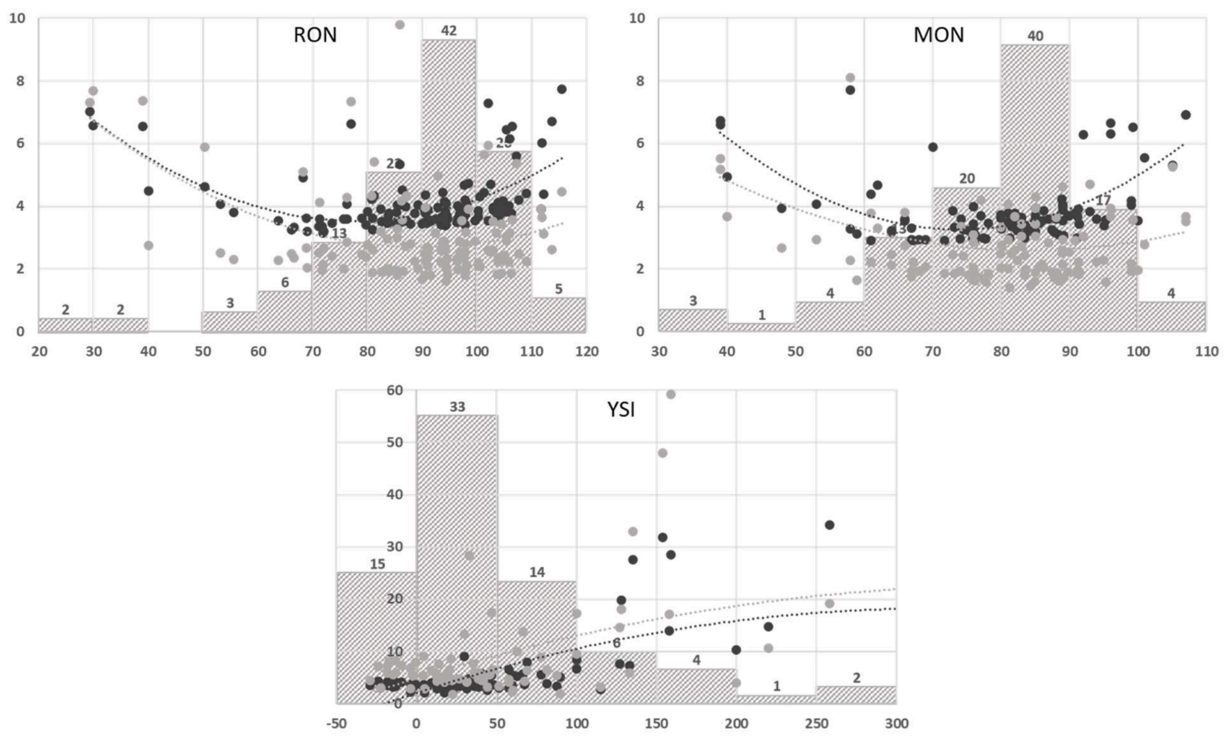


Fig. 5. Standard deviation (y-axis) vs true values (x-axis) for RON, MON and YSI of the test set. The histograms show the density of data points within the property ranges on x-axis. Gray points and gray trend line correspond to the BNN model, and black points and black trend line correspond to the Gaussian DropConnect model.

Table 4
Comparison of the performance of Gaussian DropConnect and BNN models for the test set. (MAE – Mean Absolute Error (the values in parentheses indicate MAE for each of the three properties), CI – Confidence Interval (the three values indicate the percentage of data in 1-, 2-, 3- σ CI, respectively), GNLL – Gaussian Negative Log Likelihood (the values in parentheses indicate GNLL for each of the three properties)).

Model Description	MAE (RON/ MON/ YSI)	RON CI (1-, 2-, 3- σ)	MON CI (1-, 2-, 3- σ)	YSI CI (1-, 2-, 3- σ)	GNLL (RON/ MON/ YSI)
MC Gaussian DropConnect+ Aleatoric	12.08 (3.41, 2.61, 6.04)	75%, 93%, 96%	82%, 95%, 98%	53%, 80%, 89%	6.99 (2.12, 1.81, 3.05)
Bayesian Neural Network + Aleatoric	11.97 (2.98, 2.52, 6.46)	69%, 89%, 95%	71%, 90%, 96%	72%, 89%, 93%	8.44 (1.83, 2.17, 4.43)

Data availability

The authors do not have permission to share data.

Acknowledgments

The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory (Argonne). The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in the said article to reproduce, prepare derivative works, distribute copies to the public, and perform publicly and display publicly, by or on behalf of the Government. This work was supported by the U.S. Department of Energy (DOE), Office of Science under contract DE-AC02-06CH11357. Swing, a High Performance LCRC graphics processing unit (GPU) cluster facility at Argonne National Laboratory was used for this study. The authors would like to acknowledge funding support from Aramco Americas – Detroit for this work. The work at KAUST was supported by the Saudi Aramco FUELCOM 3 program and by KAUST award number OSR-2019-CRG7-4077.

References

[1] Ihme M, Chung WT, Mishra AA. Combustion machine learning: principles, progress and prospects. Prog Energy Combust Sci 2022;91:101010.
[2] Benson SW, Buss JH. Additivity rules for the estimation of molecular properties. Thermodynamic properties. J Chem Phys 1958;29:546–72. no. 3.
[3] Drucker H, Burges CJ, Kaufman L, Smola AJ, Vapnik V. Advances in neural information processing systems. NIPS 1996:155.
[4] Roy MH, Larocque D. Robustness of random forests for regression. J Nonparametr Stat 2012;24:993–1006. no. 4.
[5] Kubic WL, Jenkins RW, Moore CM, Semelsberger TA, Sutton AD. Artificial neural network based group contribution method for estimating cetane and octane numbers of hydrocarbons and oxygenated organic compounds. Ind Eng Chem Res 2017;56(42):12236–45.
[6] vom Lehn F, Brosius B, Broda R, Cai L, Pitsch H. Using machine learning with target-specific feature sets for structure-property relationship modeling of octane numbers and octane sensitivity. Fuel 2020;281:118772.
[7] Kessler T, Sacia ER, Bell AT, Mack JH. Artificial neural network based predictions of cetane number for furanic biofuel additives. Fuel 2017;206:171–9.
[8] Abdul Jameel AG, Van Oudenhoven V, Emwas A-H, Sarathy SM. Predicting octane number using nuclear magnetic resonance spectroscopy and artificial neural networks. Energy Fuels 2018;32(5):6309–29.
[9] McClelland HE, Jurs PC. Quantitative structure−property relationships for the prediction of vapor pressures of organic compounds from molecular structures. J Chem Inf Comput Sci 2000;40(4):967–75.
[10] Li G, Hu Z, Hou F, Li X, Wang L, Zhang X. Machine learning enabled high-throughput screening of hydrocarbon molecules for the design of next generation fuels. Fuel 2020;265:116968.
[11] Saldana DA, Starck L, Mougin P, Rousseau B, Creton B. On the rational formulation of alternative fuels: melting point and net heat of combustion predictions for fuel compounds using machine learning methods. SAR QSAR Environ Res 2013;24(4):259–77.
[12] Yalamanchi KK, van Oudenhoven VCO, Tutino F, Monge-Palacios M, Alshehri A, Gao X, et al. Machine learning to predict standard enthalpy of formation of hydrocarbons. J Phys Chem A 2019;123(38):8305–13.
[13] Yalamanchi KK, Monge-Palacios M, van Oudenhoven VCO, Gao X, Sarathy SM. Data science approach to estimate enthalpy of formation of cyclic hydrocarbons. J Phys Chem A 2020;124(31):6270–6.

[14] Lazzús JA. Prediction of flammability limit temperatures from molecular structures using a neural network–particle swarm algorithm. J Taiwan Inst Chem Eng 2011;42(3):447–53.
[15] Jiao Z, Hu P, Xu H, Wang Q. Machine learning and deep learning in chemical health and safety: a systematic review of techniques and applications. ACS Chem Heal Saf 2020;27(6):316–34.
[16] Goh G.B., Siegel C., Vishnu A., Hodas N.O., Baker N. Chemception: a deep neural network with minimal chemistry knowledge matches the performance of expert-developed QSAR/QSPR models. arXiv:1706.06689. 2017.
[17] Jiang D, Wu Z, Hsieh CY, Chen G, Liao B, Wang Z, Shen C, Cao D, Wu J, Hou T. Could graph neural networks learn better molecular representation for drug discovery? A comparison study of descriptor-based and graph-based models. J Cheminform 2021;13:1–23. no. 1.
[18] Kuzhagaliyeva N., Horváth S., Williams J., Nicolle A., Sarathy S.M. Artificial intelligence-driven design of fuel mixtures. Commun Chem 5 (1), 111.
[19] Wang H, Sheen DA. Combustion kinetic model uncertainty quantification, propagation and minimization. Prog Energy Combust Sci 2015;47:1–31.
[20] Wang J, Zhou Z, Lin K, Law CK, Yang B. Facilitating Bayesian analysis of combustion kinetic models with artificial neural network. Combust Flame 2020;213:87–97.
[21] Yousefian S, Bourque G, Monaghan RFD. Bayesian inference and uncertainty quantification for hydrogen-enriched and lean-premixed combustion systems. Int J Hydrog Energy 2021;46:23927–42.
[22] Gal Y. Uncertainty in deep learning, 1. University of Cambridge; 2016 (Ph.D. thesis).
[23] Cronin MTD, Richarz AN, Schultz TW. Identification and description of the uncertainty, variability, bias and influence in quantitative structure-activity relationships (QSARs) for toxicity prediction. Regul Toxicol Pharmacol 2019;106:90–104.
[24] Zhang Y. Bayesian semi-supervised learning for uncertainty-calibrated prediction of molecular properties and active learning. Chem Sci 2019;10:8154–63.
[25] Wang D, Yu J, Chen L, Li X, Jiang H, Chen K, et al. A hybrid framework for improving uncertainty quantification in deep learning-based QSAR regression modeling. J Cheminform 2021;13(1):1–17.
[26] Mervin LH, Johansson S, Semenova E, Giblin KA, Engkvist O. Uncertainty quantification in drug design. Drug Discov Today 2021;26(2):474–89.
[27] Vishwakarma G, Sonpal A, Hachmann J. Metrics for benchmarking and uncertainty quantification: quality, applicability, and best practices for machine learning in chemistry. Trends Chem 2021;3(2):146–56.
[28] Gal Y, Ghahramani Z. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In: International conference on machine learning; 2016. p. 1050–9. PMLR.
[29] Schweidtmann AM, Rittig JG, König A, Grohe M, Mitsos A, Dahmen M. Graph neural networks for prediction of fuel ignition quality. Energy Fuels 2020;34(9):11395–407.
[30] McEnally C.S., Das D.D., Pfefferle L.D. Yield sooting index database volume 2: sooting tendencies of a wide range of fuel compounds on a unified scale (2017).
[31] Das DD, John PCS, McEnally CS, Kim S, Pfefferle LD. Measuring and predicting sooting tendencies of oxygenates, alkanes, alkenes, cycloalkanes, and aromatics on a unified scale. Combust Flame 2018;190:349–64.
[32] Zhu J, et al. Experimental and theoretical study of the soot-forming tendencies of furans as potential biofuels. Tech. rep. New Haven: Yale Univ; 2020. CT (United States).
[33] National Renewable Energy Laboratory. Co-optimization of fuels & engines: fuel properties database https://www.nrel.gov/transportation/fuels-properties-database (2018).
[34] A.S.T.M. Int. Standard test method for research octane number of spark-ignition engine fuel, 2012; ASTM D2699-21.
[35] ASTM. Int. Standard test method for motor octane number of spark-ignition engine fuel, 2011; ASTM D2700-21.
[36] Goan E, Fookes C. Bayesian neural networks: an introduction and survey. Case studies in applied bayesian data science: CIRM Jean-Morlet Chair. Fall 2018:45–87 (2020).
[37] Jordan MI, Ghahramani Z, Jaakkola TS, Saul LK. An introduction to variational methods for graphical models. Mach Learn 1999;37(2):183–233.
[38] Jospin LV, Laga H, Boussaid F, Buntine W, Bennamoun M. Hands-on Bayesian neural networks—a tutorial for deep learning users. IEEE Comput Intell Mag 2022;17(2):29–48.
[39] Chang D.T. Hybrid Bayesian neural networks with functional probabilistic layers. arXiv preprint arXiv:2107.07014 (2021).
[40] Martinez-Cantin Ruben. BayesOpt: a Bayesian optimization library for nonlinear optimization, experimental design and bandits. J Mach Learn Res 2014;15(Nov):3735–9.