Transportation Research Part C 101 (2019) 276–296

Contents lists available at ScienceDirect

Transportation Research Part C

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

Vehicle energy consumption estimation using large scale simulations and machine learning methods

Junlin Yao, Ayman Moawad*

Ecole Polytechnique, Route de Saclay, 91128 Palaiseau, France
ETH Zürich, Rämistrasse 101, 8092 Zürich, Switzerland
Argonne National Laboratory, 9700 Cass Ave, Lemont, IL 60439, USA

ARTICLE INFO

Keywords:
Vehicle simulation
Energy consumption estimation
Artificial neural networks
Machine learning
Numerosity reduction
Random sampling
Autonomie
CAFE

ABSTRACT

Vehicle energy consumption and powertrain operations for future vehicle powertrain technologies are predicted through full-vehicle simulations using Autonomie, a simulation tool developed at Argonne National Laboratory. Over 1 million vehicles, with different powertrain types (conventional and numerous electrified powertrains) and component technologies (advanced engines, batteries, electrical machines, light weighting) are simulated for outputs including vehicle fuel economy and cost. Despite the ability of Autonomie to predict fuel economy given various combination of vehicle inputs, the full-vehicle simulation is very time-consuming. It takes approximately 84 h to simulate for 33,060 vehicles even using distributed computing on a cluster of 128 worker nodes. Moreover, Autonomie is unable to populate a continuous output space through simulation since it only yields outputs with discrete values instead of an input-output relation.

This paper proposes a novel large-scale learning and prediction process (LSLPP) via machine learning approaches to answer the need for a continuous space of vehicle fuel economy and a more efficient simulation process. By learning in a supervised fashion to discover structures in data, the machine learning approaches seek to train a model that can explain the data with which we can perform prediction over unseen data. As such, we are able to accomplish analytic continuation on the current discrete space of simulation results. In addition, we present a random-sampling-based numerosity reduction algorithm incorporated in LSLPP that reduces simulation runs and improves efficiency of data procurement. We analytically derive a theoretical bound in order to guarantee the effectiveness of our algorithm. We also compare random sampling to stratified sampling and suggest sampling strategies. A user-friendly tool is developed to support the process. Our experimental results confirm that the proposed LSLPP is able to greatly accelerate prediction and analysis of vehicle fuel economy. The output of this study is used by the US government to evaluate the impact of its R&D funding on energy security and CO_2 emission as well as for the setting of Corporate Average Fuel Economy (CAFE) standards.

1. Introduction

In 1975, Congress passed the Energy Policy and Conservation Act, aiming to provide energy efficiency and reduce energy demand by imposing requirements on fuel economy, a quantity measuring the fuel efficiency and expressed in terms of the distance traveled per unit volume of consumed fuel. The Act required standards for Corporate Average Fuel Economy (CAFE), i.e., the average fuel

* Corresponding author.

https://doi.org/10.1016/j.trc.2019.02.012
Received 18 October 2017; Received in revised form 6 January 2019; Accepted 14 February 2019
Available online 06 March 2019
0968-090X/ Published by Elsevier Ltd.

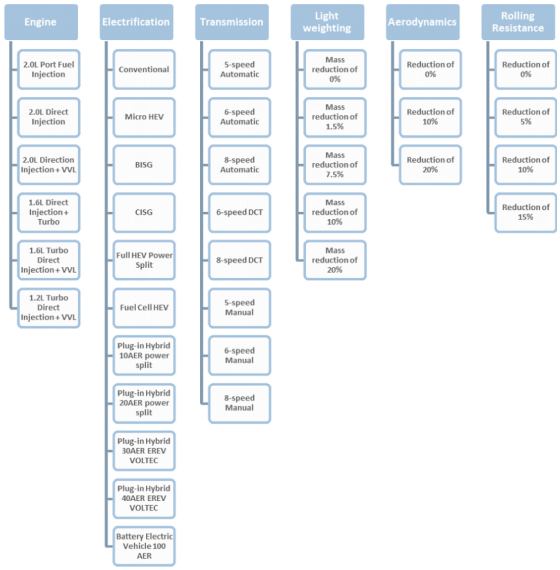

Fig. 1. Sample of all technological decision trees in the Volpe model.

economy of cars and light trucks produced for sale in the United States, and charged the U.S. Department of Transportation (DOT) with establishment and enforcement of these standards. DOT made use of the CAFE Compliance and Effects Modeling System ("Volpe model" or the "CAFE model") in developing the standards. Part of the model's function is to estimate CAFE improvements that a given manufacturer could achieve by applying additional technology to specific vehicles in its product line. DOT has also made use of vehicle simulation results to update technology effectiveness estimates by the model. The Volpe model currently relies on multiple decision trees to represent component technology options, including powertrain electrification, engine, transmission, light weighting, aerodynamics and rolling resistance, as shown in Fig. 1.

The simulation process traverses each decision tree to find the technology that should be selected next in order to provide the best fuel energy consumption improvement at the lowest cost. A full-vehicle simulation tool named Autonomie has been developed by Argonne National Laboratory, a U.S. Department of Energy (DOE) national laboratory, to support CAFE rulemakings (aut, 2014). Autonomie's ability to simulate various powertrain configurations, component technologies, and vehicle-level controls over numerous drive cycles has been used to support dozens of studies focusing on fuel efficiency, cost-benefit analysis, or greenhouse gases. Given various combination of vehicle inputs including powertrain types, engines, and other technological parameters, Autonomie is capable to yield reliable output estimates, in particular, fuel economy estimates. The main objective of the present study is to provide an efficient tool for performing individual vehicle simulations. To do so, individual vehicles have to be simulated to represent every combination of vehicle, powertrain, and component technologies.

A Large-Scale Simulation Process (LSSP) (Moawad et al., 2016) was developed by Argonne National Laboratory and implemented in Autonomie to minimize the need for decision trees and replace the synergy factors with inputs provided directly from a vehicle simulation tool. With this LSSP, running 33,060 vehicles requires more than 250,000 simulations, from sizing algorithms, imposing recurrence and iteration/looping, to vehicle simulation on cycles and combined or Plug-in Hybrid Electric Vehicle (PHEV) procedures. The total simulation time for the 33,060 vehicles was about 84 h (3.5 days) utilizing a cluster of 128 worker nodes dedicated to the System Modeling and Control Group. (Moawad et al., 2016) has also built a statistical model using the random forest method in order to conduct predictive modeling for several outputs such as Combine Fuel Consumption, Vehicle Mass, etc. In addition, preliminary analysis has been performed to reduce the necessary number of simulation runs.

Successfully as LSSP reduces traversal of decision trees in Volpe model and simulations in Autonomie, with continuously updated technologies to enrich the decision trees, simulations in Autonomie would become more and more time-consuming. A novel approach to further minimizing the need for decision trees is essentially required. It is also often that we need to study a continuous relation between vehicle inputs and outputs. However, Autonomie is unable to populate a continuous output space through simulation since Autonomie only yields outputs with discrete values. Inspired by the previous work (Moawad et al., 2016), the present report provides a description of a novel learning and prediction process via machine learning algorithms, which, to the best of our knowledge, is the first work that successfully addresses the above issues. It is worth noting that (Xu et al., 2016, 2017; Li et al., 2017) aim to provide the best fuel energy consumption improvement from the perspective of control and cruising strategies while our approach is more data-oriented.

Our work makes several contributions described below.

- We develop a large-scale learning and prediction process (LSLPP) through machine learning algorithms that incorporates preprocessing, outlier detection, training, evaluation, prediction and analysis for all powertrain types and all corresponding outputs. As such, we are able to perform prediction over unseen data and accomplish analytic continuation on the current discrete space of

simulation results.
- Moreover, we present a numerosity reduction algorithm via random sampling to reduce simulations in Autonomie. We guarantee its effectiveness by analytically deriving a theoretical bound.
- We also design a MATLAB-based learning and prediction tool to support the proposed process.

2. Overview of large-scale learning and prediction process

LSLPP seeks to establish analytic continuation on the current discrete space of simulation results and minimize simulation runs in Autonomie. For the purpose of clarity, the process can be divided into two routines depending on the specific objective that each routine achieves: (a) machine learning and prediction, and (b) numerosity reduction. The machine learning and prediction routine explores algorithms that can learn from data (Kohavi and Foster, 1998), generalize regularities implicit in data, and make data-driven prediction over unseen data. The numerosity reduction routine refers to parametric or nonparametric techniques that consist in reducing the data volume (Han et al., 2006) by choosing a representative form of data, which in our case leads to reducing simulations and making data procurement more efficient. In other words, the two routines can be considered as data exploration and data preparation, which are closely interconnected. On one hand, numerosity reduction prepares a representative dataset from which machine learning algorithms can efficiently learn. On the other hand, patterns in data discovered by machine learning techniques can in turn inspire numerosity reduction to select appropriate data as we will describe in Section 4. Since numerosity reduction makes use of the framework of the data exploration, we first present the machine learning and prediction routine in Section 3. We then introduce and analyze our numerosity reduction routine in Section 4.

3. Machine learning and prediction

Due to the need of continuous relation analysis, we seek to predict outputs including fuel economy based on continuous space of inputs constructed by varying input values, such as vehicle glider mass, engine power, etc., in their respective valid ranges. Since Autonomie uses physics-based mathematical equations, engineering characteristics (e.g., engine maps, transmission shift points, and hybrid vehicle control strategies), and explicit drive cycles, there surely exist general regularities in the data that can be discovered and generalized. In extending the general regularities to new data assumed to be generated by the same mechanism, we are able to predict outputs with respect to infinite input combination with continuously varying values. This is exactly how machine learning works. Supervised machine learning algorithms are trained on a well-prepared dataset so that knowledge hidden in the data can be explored and encoded in a model. Feeding new inputs to pre-trained models consequently yields predicted outputs. Even though the training routine takes time, particularly for large training set, it is far less time-consuming than simulation in Autonomie. The most remarkable benefit of machine learning consists in the fact that pre-trained models can almost instantaneously predict outputs with high accuracy. Simulation that lasts several days in Autonomie is now greatly reduced to a few minutes at the most.

Our machine learning and prediction routine is composed of steps shown in Fig. 2: preprocessing, training, validation, prediction, and outlier detection if necessary. Before we present each step in detail, it should be noted that even though the main routine seems linear, it is understood to be an iterative process. The need for improvement of prediction accuracy can recall any step previous to outputting trained models. For instance, if the validation on test set implies an inappropriate model, an alternative machine learning algorithm may be selected. When outliers are assumed to be present, outlier detection process may be required. If a training set is considered insufficient, more data should be procured. In the following, we describe our machine learning and prediction routine step by step.

3.1. Preprocessing

Preprocessing aims to convert original datasets into those with appropriate formats ready for training, including dividing a miscellaneous collection of data into separate homogeneous datasets, feature selection, creating dummy variables for categorical ones, normalization, etc. The original vehicle database is structured to be generic so that any simulation input parameter, result, or descriptive property can be stored. Typical inputs and outputs are shown in Table 1. We refer readers to Appendix A for the complete list of inputs and outputs and the mapping to abbreviated variable names used in our framework.

3.1.1. Creating homogeneous datasets

While some parameters, such as vehicle class, are shared by all vehicles, others are only available to vehicles with certain powertrain types. For example, PHEV fuel economy CS on UDDS belongs only to PHEV powertrain. Consequently, the database containing vehicle data corresponding to 10 different powertrain types (conventional, BEV200, EREV PHEV30/50, fuel cell HEV, par/split HEV, micro hybrid, and mild hybrid BISG/CISG) can be viewed as a miscellaneous collection of data. Vehicle data with same inputs and outputs, or equivalently, sharing the same powertrain type, need to be grouped together to form a homogeneous training set for each powertrain type. We thus obtain 10 vehicle datasets, each of which contains a unique powertrain type.[1,2]

[1] It should be noted that this is only an intermediate step for data preparation. Prediction on a mixture of powertrain types will be finally merged though prediction is technically performed on individual powertrain types.

[2] If not explicitly stated otherwise, vehicle datasets will stand for the homogeneous datasets in what follows.

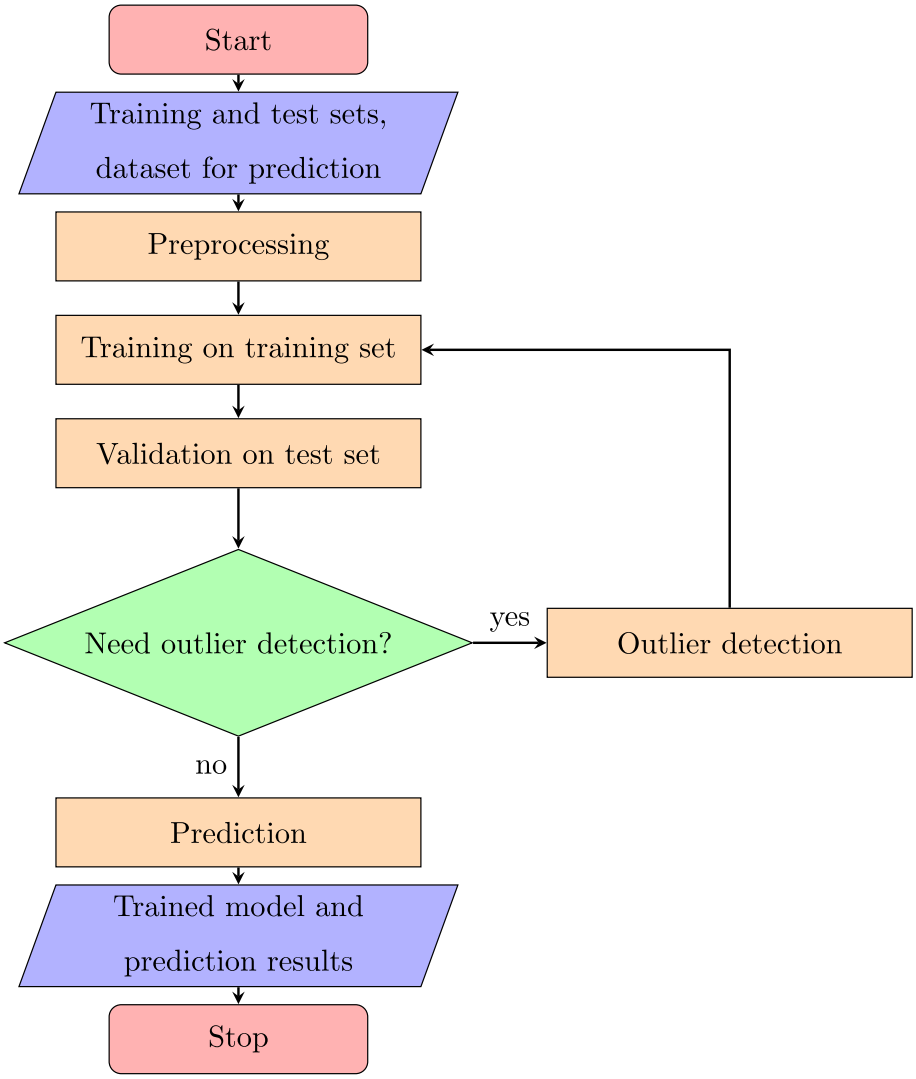

Fig. 2. Machine learning and prediction routine.

Table 1
Example of inputs and outputs in the original vehicle database.

Inputs	Outputs
• vehicle class • vehicle powertrain • engine • fuel cell max power	• vehicle curb weight • vehicle passing time • unadjusted PHEV fuel economy CS on UDDS • unadjusted fuel economy UDDS

3.1.2. Feature selection

Feature selection is the process of selecting a subset of relevant variables for use in construction of machine learning models. The main purpose of feature selection is to remove redundant or irrelevant features while preserving sufficient information in order to reduce over-fitting as well as training time. In our vehicle database, not all features are useful and necessary for model construction. For example, engine-related parameters such as engine cylinder configuration, number of banks, engine displacement, can be uniquely determined once engine type is specified. We can thus eliminate 10 redundant engine features. For each powertrain type, columns of variables that are not available are also deleted. Similar to Moawad et al. (2016), we conduct correlation analysis using the Pearson product-moment correlation coefficient (Bishop, 2006) to further remove highly correlated features such as vehicle test weight and gross vehicle weight. Through feature selection, we achieve simplification of models. In consequence, the number of inputs decreases from more than 30 to less than 10 in the case of conventional vehicles. Since each powertrain type generally has a

variableName	usage	isDummy
Vehicle_Class	1	1
Vehicle_Powertrain	1	1
Engine_IAV_Type	1	1
Engine_Fuel_type	0	1
Engine_Cylinder_Configuration	0	1
Engine_nb_of_Cylinders	0	0
Engine_Displacement	0	0
Engine_has_Cylinder_Deac	0	1
Engine_nb_of_Banks	0	0
Engine_Cam_Shaft_Configuration	0	1
Engine_Injection_Type	0	1
Engine_Valvetrain_Type	0	1
Engine_EGR_Type	0	1
Engine_has_Turbo	0	1
Engine_Boost_Level	0	0
Engine_Max_Power	1	0
Max_Toque_of_Pwt	0	0
Transmission_Type	0	1
Transmission_nb_of_gears	0	1
Glider_Mass_Reduction_Step	0	1
Glider_Mass_Reduction_Percentage	0	0
Glider_Mass	1	0
Aerodynamics_Reduction_Step	0	1
Aerodynamics_Reduction_Percentage	0	0
Drag_Coefficient	1	0
Rolling_Resistance_Reduction_Step	0	1
Rolling_Resistance_Reduction_Percentage	0	0
Rolling_Resistance	1	0
Vehicle_Test_Weight	0	0
Vehicle_Curb_Weight	2	0

Fig. 3. Data dictionary for split HEV vehicles.

different configuration of inputs and outputs, feature selection yields different remaining variables for respective powertrain types.

Results of feature selection are stored in a specially designed structure, named data dictionary. An exemplary data dictionary, shown in Fig. 3, contains three columns, namely, variable name, usage, and indicator of categorical variables.

Values in the second column indicates usage of variables, mapping input/output/unused to numerical values (Table 2). Variables marked as unused are those considered as redundant or irrelevant and thus removed.

The third column in the data dictionary suggests whether a variable is categorical and need be made “dummy”: 0 indicates a numeric; 1 indicates a categorical variable. This indicator helps convert a categorical variable to a dummy one as explained in the following.

3.1.3. Creating dummy variables

As another important step in preprocessing, we convert categorical variables to dummy ones. In general, machine learning algorithms can only handle numeric data. Categorical variables in the database such as vehicle class with values being “compact”, “pickup”, etc., should be converted to numeric values. However, it is meaningless to compare the resulting numeric values. To illustrate that, let us suppose that compact vehicle class is mapped to 3 and pickup to 5. It would be ridiculous to claim that a pickup vehicle is greater than a compact vehicle by 2. After all, the origin of the numeric is categorical. The conversion should preserve categorical effect. In statistics, a common practice is to create dummy variables that take the value 0 or 1 to indicate the category to which a variable belongs (Garavaglia and Sharma, 1998). Therefore, given a categorical variable that has two unique values, for example, “compact” and “pickup”, we convert it to a boolean dummy variable that indicates compact vehicle if it is 1 and pickup vehicle if 0.

3.1.4. Normalization

Normalizing inputs can make training process of some machine learning methods more efficient, as is the case of neural network that we use in our work. As sigmoid transfer functions are used in the hidden layers of multilayer feedforward neural networks, they tend to become saturated when the input is very large, leading to small gradients and thus slow training processes. It is common practice to normalize inputs to either fall in the range $[-1, 1]$ or have zero means and unity variance. Since values of the vehicle inputs vary greatly, we map vehicle inputs to $[-1, 1]$ using the following algorithm:

$$z = \frac{2(x - x_{\min})}{x_{\max} - x_{\min}} - 1,$$

Table 2
Mapping usages to numeric values.

Usage	Value
Input	1
Output	2
Unused	0

where x is an arbitrary input with $x_{\min}$ and $x_{\max}$ being its minimum and maximum value, respectively. As such, we obtain the normalized input z.

The preprocessing ends by splitting an entire dataset into two non-overlapping sets, namely, training set and test set, in order to prevent over-fitting and to properly evaluate prediction accuracy. Typically, a training set contains 70% of data points and a test set contains the complementary subset. Machine learning models are trained on the training set while validation is performed on the test set.

3.2. Training

The learning step, also known as training, refers to the process where a supervised machine learning algorithm fits a model, sometimes called a learner, to a training set. Different algorithms behave differently, yielding dissimilar models and prediction performance. In our study, we utilize multilayer feedforward neural network models (Svozil et al., 1997), a state-of-the-art machine learning approach for nonlinear regression, due to the fact that relations between inputs and outputs are generally non-linear. In addition, the ability of multilayer feedforward neural networks to approximate any measurable function to any desired degree of accuracy makes them universal approximators (Hornik et al., 1989) and hence a good candidate for our LSLPP.[3]

We can accelerate the training process using parallel computing. It takes approximately 1–5 min to train a neural network for predicting an output using 6 processors. It is worth noting that our machine learning and prediction process is not restricted to any machine learning algorithm. Other approaches for regression such as random forest can also be integrated to our process as learners.

3.3. Validation and evaluation

When fitting models to a training set, we may lose generality while focusing on specific patterns, resulting in unreliable prediction. This common problem is referred to as over-fitting. In order to find a trade-off between generality and specifics, we assess models using cross-validation techniques.

We first test the pre-trained model on a separate dataset, namely the test set generated during preprocessing. More specifically, given inputs of the test set, the machine learning model predicts an output which is compared to the ground truth, i.e., the correct values of the output provided in the test set. For the regression task, the mean square error (or mean squared error, MSE), is computed to measure the prediction residuals as follows:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2,$$

where n is the number of data points in the test set, and $\hat{Y}_i$ and Y_i are the predicted value via the machine learning model and the ground truth obtained by Autonomie, respectively.

Although validation is integrated into the training process of neural networks to avoid over-fitting and trigger early stopping, we still assess performance of trained models on the test set. A model that achieves low MSE is reliable to be used to predict unseen data. For illustration, Fig. 4 shows the evaluation of the model for predicting unadjusted BEV electrical energy consumption on UDDS drive cycle. The neural network model achieves a very low MSE, i.e., 0.02833, a correlation coefficient close to 1, and very few errors outside the error bar (a 99.3% coverage if data are assumed to be normally distributed). We can thus conclude that the model successfully discovers implicit patterns in the data and reliably generalizes the regularities to unseen data.

The above method (also called *holdout method*) can be improved by carrying out several rounds of validation instead of one single round, namely, the *k-fold* cross-validation. As its name indicates, we randomly partition the original dataset into k equal sized subsets. During each round, $k-1$ subsets are used as the training set with the remaining one as the test set, which can be basically considered as the holdout method. The k-fold cross-validation consists in repeating k times the single holdout validation with each subset being used to form the training set only once, and averaging k results to produce a single estimate. Consequently, we reduce the impact of randomness caused by the partition and random initialization of weights in the neural networks. In our validation process, we use 10-fold cross-validation to evaluate models for all powertrain types and all outputs.

We present validation results as shown in Table 3 for illustration. The average MSE, denoted by $\overline{\mathrm{MSE}}$, as well as the standard deviation of MSE, denoted by σ_{MSE}, are computed. We also measure the differences between predicted outputs and outputs in ground truth, or the targets, through RMSE, defined as the square root of MSE, which has the same unit as the target. Similar to the average MSE, we calculate the average RMSE, $\overline{\mathrm{RMSE}}$. Finally, we normalize the RMSE in order to help compare prediction performance for outputs with different scales. By dividing the mean of the output, $\overline{Y}$, by $\overline{\mathrm{RMSE}}$, we obtain a measure of the relative mean error independent of scales, known as the *coefficient of variation of the* $\overline{\mathrm{RMSE}}$ or CV($\overline{\mathrm{RMSE}}$), expressed as a percentage.

The above cross-validation result of conventional vehicles justifies the reliability of the neural network models. Both the average MSE and its standard deviation are low compared to the values of the outputs. The normalized RMSE is less than 0.8% for all outputs. The results for other powertrain types can be found in Appendix A.

We summarize the results of validation and evaluation for all vehicle datasets in what follows.

- Conventional, BEV200, EREV PHEV30/50, fuel cell HEV, micro hybrid, mild hybrid BISG, par/split HEV: good fitting, low MSE,

[3] We use the multilayer feedforward neural network as our main machine learning algorithm in what follows.

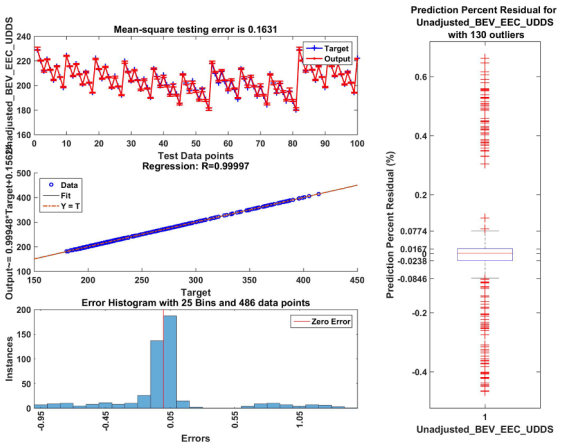

Fig. 4. Evaluation of the model for predicting unadjusted BEV electrical energy consumption on UDDS drive cycle.

Table 3
Cross-validation results of conventional vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	0.0346	0.0123	0.184	1510	0.0121
Vehicle_Acceleration_Time	0.00253	0.000611	0.05	9.19	0.544
Vehicle_Passing_Time	0.00227	0.000178	0.0476	6.71	0.71
Unadjusted_FE_UDDS	0.00626	0.000799	0.0789	34.6	0.228
Unadjusted_FE_HWFET	0.00931	0.00235	0.0958	48.9	0.196
Unadjusted_FE_US06	0.00436	0.000404	0.066	29.9	0.221

correlation coefficient close to 1, very few errors outside the error bar;

- Mild hybrid CISG: good fitting, low MSE, correlation coefficient close to 1, very few errors outside the error bar except for fuel economy on UDDS with relatively lower prediction accuracy.

Therefore, multilayer feedforward neural networks are capable of making solid predictions for most of outputs. As for fuel economy on UDDS of hybrid CISG vehicles, we believe that the relatively low prediction accuracy may be due to the fact that excessive charging or discharging occurs during drive cycle. Since currently no variable in datasets measures and reflects the state of charge (SOC), our machine learning model lacks information to make accurate predictions. By incorporating a new variable providing such information, the machine learning algorithm is expected to yield satisfactory prediction. Since hybrid vehicles are still under simulation when the paper is written, data corresponding to hybrid vehicles are excluded from vehicle datasets in what follows.

3.4. Outlier detection

Among factors affecting prediction accuracy is the existence of outliers in training sets. Despite no universally accepted definition for an outlier, we will take the definition of Barnett and Lewis (1994) which states that an outlier can be defined as the data point that appears to be inconsistent with the remaining ones in the same dataset. While Autonomie generally yields reliable simulation results, some potential issues such as too many shifting events on a specific cycle (Moawad and Rousseau, 2014) may lead to abnormal outputs, or outliers. Training machine learning models on a dataset contaminated by outliers would result in an unworkable model.

In order to avoid the above issue, we incorporate random sample consensus algorithm (RANSAC) (Fischler and Bolles, 1981), an outlier detection technique, in our machine learning and prediction process. By assuming that outliers are data that do not fit the underlying model shared by most data points, RANSAC is capable to estimate in an iterative way the best model that fits sufficiently many points supposed to be inliers. Its ability to make robust estimation of the model parameters allows us to obtain high prediction accuracy even with the presence of a significant number of outliers in the dataset.

For illustration, let us consider the prediction of unadjusted fuel economy on UDDS drive cycle of fuel cell HEV. Figs. 5 and 6 show evaluation of the models trained on the entire dataset and the cleansed dataset via outlier detection, respectively. In spite of the overall high accuracy for the first model, we observe that some data points have extreme prediction errors, implying the existence of outliers. We thus conduct outlier detection process and train second model on the cleansed dataset without abnormal data points. Compared to the first model, the second one makes more accurate prediction with MSE being 0.00044, much lower than 0.0017, the MSE corresponding to the first model. In addition, even though some prediction residuals are still out of the error bar as shown in the

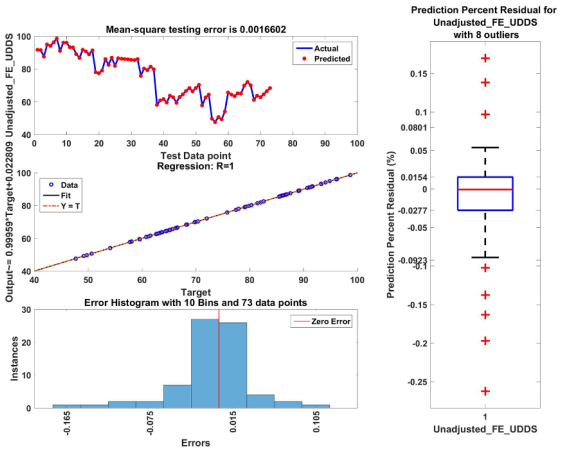

Fig. 5. Evaluation of the model for predicting unadjusted fuel economy on UDDS drive cycle of fuel cell HEV using the entire dataset.

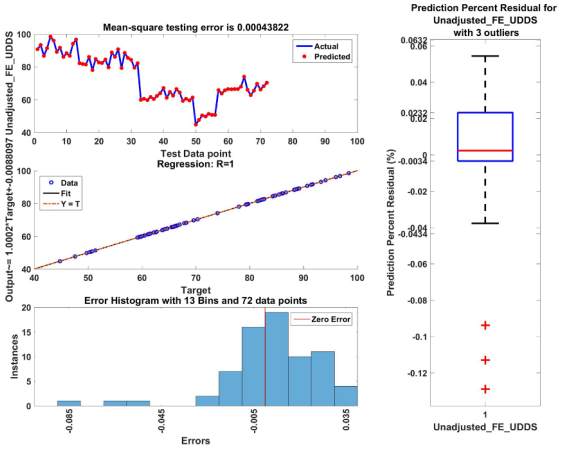

Fig. 6. Evaluation of the model for predicting unadjusted fuel economy on UDDS drive cycle of fuel cell HEV using the cleansed dataset after outlier detection.

box plot in Fig. 6, their deviation from inliers is much less than that in the case of contaminated dataset as shown in Fig. 5. This demonstrates the effectiveness of the outlier detection process.[4]

3.5. Prediction

Recall that during validation and evaluation, we feed inputs to models in order to predict vehicle outputs. The prediction step works similarly except that no ground truth is provided to evaluate prediction. By continuously varying inputs, we are able to accomplish analytic continuation on the current discrete space of simulation results.

In addition to predicting outputs, it is often desirable to quantify the accuracy of prediction with respect to the ground truth. In other words, we need to estimate the *confidence interval* of predicted values from a multilayer feedforward neural network. Various approaches exist for this estimation task, including Jacobian-based estimators (Chryssolouris et al., 1996), bootstrap estimators (Paass, 1993), etc. The bootstrap method has been proved to outperform other estimators due to its capacity to approximate a sampling distribution of the prediction without making parametric assumption about the distribution and to successfully capture randomness of neural networks (Tibshirani, 1996). Therefore, we apply the bootstrap to estimation of confidence interval for feedforward neural networks.

Assume that $Z(n) \coloneqq (z_1, \ldots, z_n)$, where z_i refers to a pair of input and output variables (x_i, y_i), with x_i being a vector of input variables and y_i a scalar output variable. Let v be a vector of input variables with unknown output values for prediction. We follow the *pairwise bootstrap* algorithm proposed by Efron (1982) to estimate confidence intervals.

[4] The number of data points in the test set of RANSAC-based training (72) is slightly smaller that of basic training (73) due to different rounding algorithms in MATLAB.

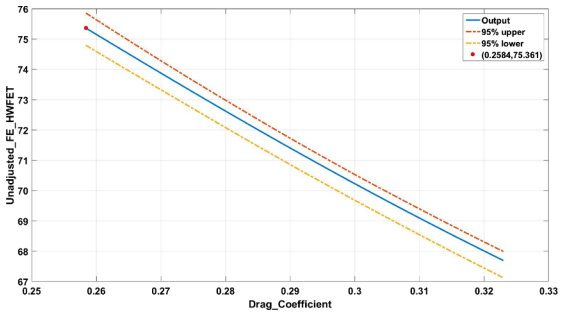

Fig. 7. Prediction of unadjusted fuel economy HWFET of split HEV compact vehicles is guaranteed by a 95% confidence interval (upper and lower bounds in red and yellow, respectively) using the bootstrap technique. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

1. We randomly draw a sample $Z_b^*(n)$ of size n with replacement from the training set $Z(n)$ for $b = 1, ...,B$.
2. A feedforward neural network is trained on bootstrap samples $Z_1^*(n), ...,Z_B^*(n)$ and we obtain an ensemble of B trained neural networks.
3. v is fed to B neural networks which yield B predicted output values, $o_1, ...,o_B$.
4. We take the average output $\bar{o} = \frac{1}{B}\sum_{i=1}^{B} o_i$ as the ultimate prediction given the input v.
5. Let $\widehat{F}_B$ denote the empirical *cumulative distribution function* (cdf). Then, the approximate $(1 - 2\alpha)$ central confidence interval of $\bar{o}$ is given by $[\widehat{F}_B^{-1}(\alpha), \widehat{F}_B^{-1}(1 - \alpha)]$.

In practice, we use the $100p^{th}$ percentile $(0 < p < 1)$ to approximate $\widehat{F}_B^{-1}(p)$, which provides reliable estimates of confidence intervals. As shown in Fig. 7, we predict the unadjusted fuel economy HWFET of split HEV compact vehicles where we obtain the continuous relation of fuel economy HWFET versus drag coefficient by varying drag coefficient from 0.2584 to 0.323. [5] The prediction is guaranteed by a 95% confidence interval (upper and lower bounds in red and yellow, respectively) using the bootstrap technique.

3.6. Machine learning and prediction tool

With the established machine learning and prediction routine, we design a MATLAB-based machine learning and prediction tool that integrates preprocessing, training, validation, prediction, and outlier detection process. Moreover, the tool also provides other options of machine learning algorithms, such as bagged trees. With the help of our tool, users are able to conduct analysis in a flexible yet straightforward way. For instance, users are able to analyze continuous relation between vehicle inputs and fuel economy using the real time prediction mode interface provided by the tool (see Fig. 8).

The features of our machine learning and prediction tool are summarized as follows.

- Provide generic data-independent interface to guide users to perform the machine learning and prediction routine.
- Offer two prediction modes: by-batch prediction (grouped inputs) or real-time prediction (inputs defined by users in real time).
- Allow users to visualize and evaluate model performance.
- Enable users to detect and remove abnormal data (outliers) in order to improve prediction accuracy.
- Optimize training on large datasets using paralleling computing.

In the future, we expect to provide an off-the-shelf tool that is extended from the current MATLAB-based tool for users to predict and analyze vehicle fuel economy on various drive cycles in addition to the standard drive cycles presented in this paper.

4. Numerosity reduction

In this section, we introduce our numerosity reduction routine to make data procurement more efficient, namely, the large-scale simulation in Autonomie. We fulfill this objective using random sampling. By applying machine learning algorithms to samples randomly drawn from the entire dataset, we are able to run simulations more efficiently while still achieving high prediction accuracy. We also address the issue caused by an unbalanced random sampling that misses some representative data points in the presence of clusters in the dataset. We give an upper bound to the probability that random sampling fails to capture certain clusters and analytically derive a minimum sample size given a desired upper bound. Furthermore, we compare random sampling to stratified sampling and confirm the performance of our method through various experiments. Finally, we summarize our sampling strategies.

[5] The engine type is Prius 2010 with engine power being 74.215 kW. Other parameters related to technological improvement such as glider mass and rolling resistance are set to be 656 kg and 0.006, respectively. The motor 1 max power is 52.652 kW.

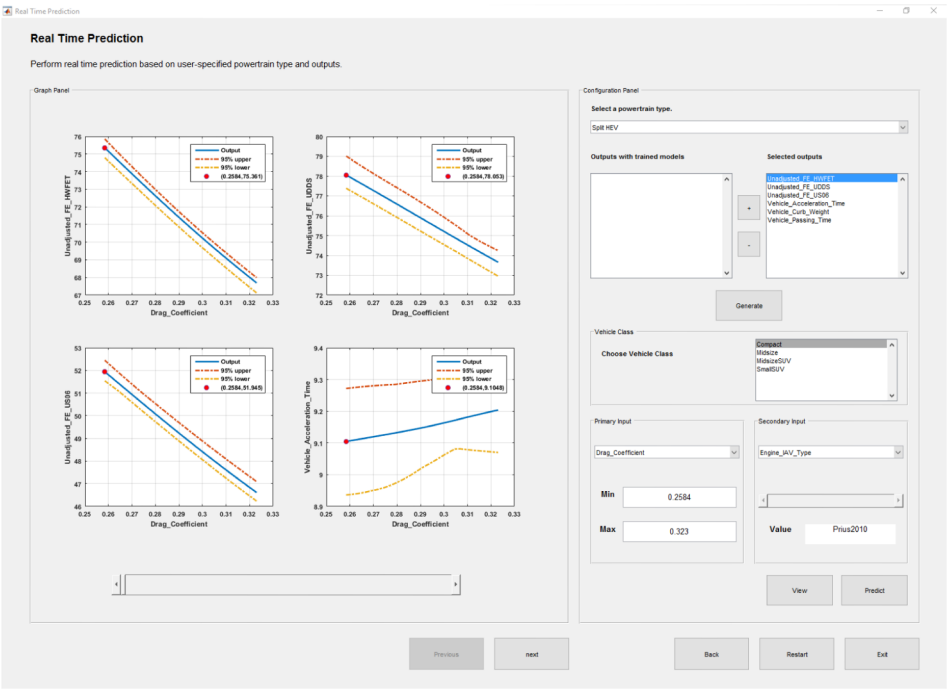

Fig. 8. Machine learning tool: use real time prediction mode to analyze continuous relation between inputs and outputs.

4.1. *Random-sampling-based numerosity reduction*

In general, the more training points are used, the higher prediction accuracy the machine learning methods can obtain. However, as suggested in Moawad et al. (2016), we observe in the experiments that the original dataset contains redundancy. Further increase in the number of training points does not reduce MSE significantly. In other words, only a subset of data points is sufficient for a machine learning algorithm to capture general regularities implicit in the data and predict outputs with high accuracy. By sampling a fraction of representative data points, we expect to efficiently procure a training set while effectively generalizing simulation outputs by means of machine learning approaches.

In order to analyze impact of sampling fraction, i.e., the ratio of sample size to population size, on prediction accuracy, we conduct experiments as follows. Given a vehicle dataset, we set aside a test set (10% of the entire dataset) for independent evaluation. We then sample a fraction of the remaining data to construct a training set. By varying sampling fraction from 5% to 90%, we are able to obtain training sets with different sizes, from which a multilayer feedforward neural network learns patterns and relation. As an accuracy measure, MSE is calculated on the test set. Moreover, we repeat the experiments for a total of 50 repetitions in order to reduce the effects of the randomness from the neural network and the random sampling.

For illustration, we consider unadjusted fuel economy HWFET of conventional vehicles though the result can be generalized to

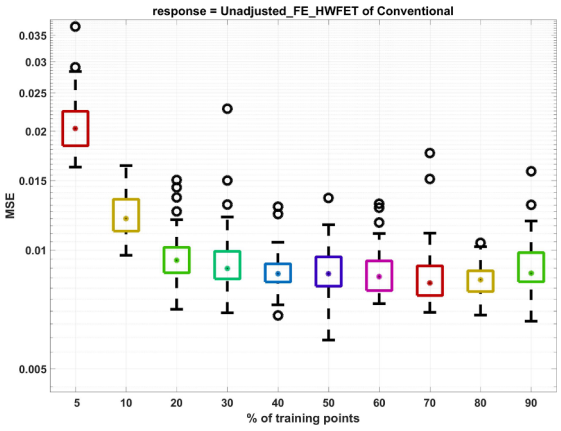

Fig. 9. MSE as a function of the sampling fraction obtained over 50 repetitions for unadjusted fuel economy HWFET of conventional vehicles.

Table 4
Recommended sampling fractions for all powertrain types.

	# of data points	Sampling fraction (%)
Conventional	61,236	**30–40**
BEV200	486	70–80
EREV PHEV30	432	40–50
EREV PHEV50	432	40–50
Fuel Cell HEV	486	70–80
Par HEV	6102	**30–40**
Split HEV	432	60–70

other powertrain types and outputs. Fig. 9 shows the set of box plots of MSE obtained over 50 repetitions for increasing sizes of training sets. We observe that in general the model can achieve lower MSE as more points are used in training sets. Nevertheless, further increase in the number of training points does not reduce MSE significantly. For a sampling fraction being 30%, the neural network algorithm can achieve high prediction accuracy (MSE is less than 0.01) which is very close to the one obtained when 90% of points are used.

Our experiments on other powertrain types and outputs confirm that for sufficiently large datasets, using a small subset of the dataset via random sampling is enough to build a model with satisfactory performance. Based on our experimental results, we are able to determine a sample size n that satisfies a desirable trade-off between efficiency and accuracy. Since we need predict several outputs for each powertrain types, after selecting desirable sample sizes $n_1, n_2, \ldots, n_m$ for m outputs of this powertrain type through experiments, we obtain a universal sample size n^* that is applied to the entire vehicle dataset regardless of outputs by finding the maximum of individual sample sizes, i.e., $n^* = \max_{1 \leqslant i \leqslant m} n_i$. This ensures that we choose a sampling fraction that satisfies all outputs given a powertrain type. The recommended sampling fractions for all powertrain types based on our numerosity reduction routine are given in Table 4.

Our experimental results also imply that we need only 30–40% of data points for large datasets, namely, datasets of conventional and par HEV powertrain types. On the other hand, we require greater sample fraction ranging from 40% to 80% for small datasets in order to ensure a sufficient representative sample.[6] In total, we are able to reduce simulations approximately by 45,000, or 64% of simulations through simple calculation. As such, by means of our numerosity reduction routine, we succeed in improving efficiency of data procurement.

4.2. Theoretical analysis on random sampling

A representative dataset is necessary in order to construct a machine learning model that can generalize beyond observed data. On the contrary, a biased training set would lead to useless models. In our dataset, each powertrain type has various vehicle classes, such as compact, pickup, small SUV, etc. Our clustering analysis suggests that similar vehicle classes form a cluster as shown in Fig. 10, the detail of which will be presented in Section 4.2.1.

In order to illustrate the undesirable consequence that random sampling misses certain cluster, let us first consider an extreme scenario where random sampling "unfortunately" draws all points from one single cluster formed by compact BEV200 vehicles. Fig. 11 shows two sets of box plots of the MSE obtained over 50 repetitions for all outputs of BEV200, including vehicle curb weight, acceleration/passing time, electrical energy consumption on UDDS/HWFET/US06, and range on UDDS/HWFET. The left figure refers to the prediction results obtained by a biased sample containing only compact BEV200 vehicles. The right figure shows the prediction results obtained by a balanced sample containing BEV200 vehicles of all classes (compact, midsize, midsizeSUV, pickup).

Compared to a balanced random sampling that takes into account all clusters in the dataset, missing certain clusters would lead to an unacceptable MSE, making the machine learning model unworkable in practice.

We can thus assume that if there exist several clusters, a sample should contain sufficient representative data points from each cluster. The assumption can be further formulated as follows. If the number of data points belonging to the cluster c in the sample exceeds αK, where K is the number of data points in c and $\alpha \in [0, 1]$, then the machine learning approach that is applied to the sample dataset can successfully learn and predict with low MSE in the case of regression. The above assumption can be justified since in general, clusters have high intra-cluster densities and low inter-cluster densities. Points within the same cluster are thus compact and similar to a representative core set in the cluster. We develop the assumption in more details in the following.

4.2.1. Clustering analysis

We formally conduct clustering analysis to discover clusters in each vehicle dataset using k-medoids algorithm (Kaufman and Rousseeuw, 1987). Similar to k-means algorithm, k-medoids clustering algorithm divides the dataset into groups by minimizing the distance between a center of a cluster and others points labeled to be in that cluster. k-medoids algorithm, however, is able to measure distances with any arbitrary metrics, unlike k-means which only works with ℓ_2 (Hartigan and Wong, 1979). Moreover, k-medoids designates points as centers of clusters while k-means uses means of points. Also, k-medoids can handle both categorical and

[6] The numerosity reduction routine is also implemented in our machine learning and prediction tool.

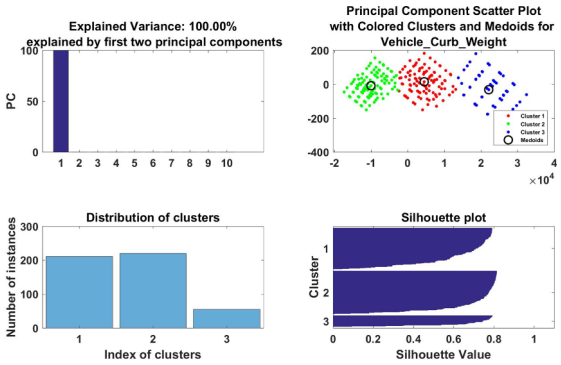

Fig. 10. Clustering result of BEV200: obtain three clusters corresponding to small-sized, middle-sized, and large-sized vehicles by applying *k*-medoids algorithm (top right) to first two principal components (PC) of inputs (top left). The silhouette method confirms that there exist three clusters (bottom right). The number of instances in each cluster is also shown, respectively (bottom left).

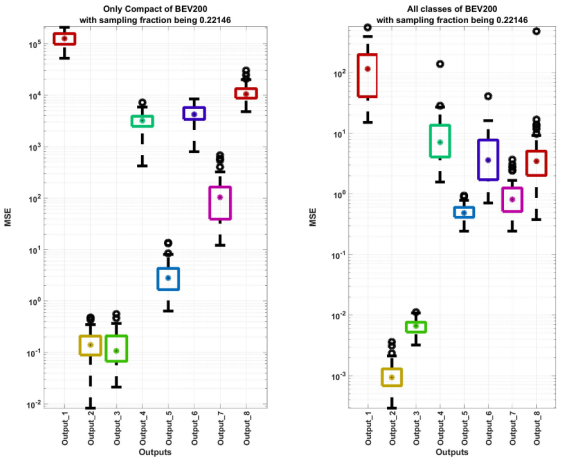

Fig. 11. Comparing prediction accuracy of biased sample with balanced sample: we draw the same number of data points from compact vehicle class (left) and from all classes (right) of BEV200, respectively. For each output we obtain much lower MSE for the balanced sample than the biased one.

numerical variables, which meets our need since the vehicle datasets contain mixed numeric and categorical data.

To make clustering more efficient, we first perform dimensionality reduction using *principal component analysis* (PCA) (Abdi and Williams, 2010). It turns out that the first two principal components can generally explain more than 99% of variance for all vehicle datasets, from which we deduce that it is sufficient to use the first two principal components. Then, we apply *k*-medoids algorithm to all vehicle datasets transformed through PCA and determine the number of clusters through silhouette, a method of validation of consistency within clusters of data. Table 5 shows that there are at least two clusters in all vehicle datasets. The presence of multiple clusters in the data could result in biased random samples, as mentioned previously. We will show in Section 4.2.2 that by drawing sufficient data points we are able to prevent undesirable samples with high probability.

4.2.2. Minimum sample size and theoretical bound

Intuitively, random sampling seeks a trade-off between accuracy and efficiency. A large sample size is more possible to avoid

Table 5
Clustering results for all vehicle datasets.

	# of clusters
Conventional	4
BEV200	3
EREV PHEV30	2
EREV PHEV50	2
Fuel Cell HEV	4
Par HEV	4
Split HEV	4

missing certain cluster while achieving lower efficiency. On the contrary, a small sample size greatly relieves burden of procurement of training set at the cost of more probably missing clusters. To quantify the above trade-off, we desire to find a theoretical bound that ensures a balanced random sampling with high probability. We intend to derive a minimum sample size given an upper bound of probability that random sampling misses certain cluster. Recall the assumption that we propose previously that the number of data points belonging to the cluster c in the sample should exceed a fraction of data points in c. We formally formulate the assumption as follows. Let X denote the random variable representing the number of data points in the sample belonging to the cluster c. Let δ be an upper bound of probability, where $0 \leqslant \delta \leqslant 1$. Then the probability that data points from c in the sample are fewer than a fraction of data points in c, which is αK, can be written as follows.

$$\mathbb{P}(X \leqslant \alpha K) \leqslant \delta, \tag{1}$$

Since we consider performing random sampling without replacement, X follows the hypergeometric distribution. More specifically, the probability that the number of data points from c in the sample exactly equals k is given by

$$\mathbb{P}\left(X = k\right) = \frac{\binom{K}{k}\binom{N-K}{n-k}}{\binom{N}{n}}, \tag{2}$$

where N is the total number of data points in the dataset, K the number of data points in the cluster c of interest, n the number of points in the sample and k the number of points belonging to c in the n sample points. Then, Theorem 1 suggests sample sizes so that with high probability random sampling will not miss certain cluster.

Theorem 1. *Set $p \coloneqq K/N$. Given a cluster c, Eq. (1) holds if the sample size n satisfies*

$$n \geqslant \frac{1}{4p^2}\left(4p\alpha K + \log\frac{1}{\delta} + \sqrt{8p\alpha K\log\frac{1}{\delta} + \left(\log\frac{1}{\delta}\right)^2}\right). \tag{3}$$

To prove the above Theorem, we need a useful conclusion on upper bounds on the tail of the hypergeometric distribution.

Theorem 2. *Suppose we have N data points in the dataset, K of which belong to the cluster c. We randomly draw n data points from the dataset without replacement. Let X denote the number of data points from c in the sample. Set $p \coloneqq K/N$. Then, for any $t \in [0, 1-p]$, we have*

$$\mathbb{P}(X \leqslant (p-t)n) \leqslant \exp(-n\cdot D_{\mathrm{KL}}(p-t||p)) \leqslant \exp(-2t^2n), \tag{4}$$

where D_{KL} refers to Kullback-Leibler divergence with

$$D_{\mathrm{KL}}\left(a\middle\|b\right) = a\log\frac{a}{b} + \left(1-a\right)\log\frac{1-a}{1-b}.$$

Theorem 2 implies that the probability that the number of data points in the sample belonging to the cluster c is no greater than $(p-t)n$ is bounded from above by $\exp(-n\cdot D_{\mathrm{KL}}(p-t||p))$ which can be relaxed to $\exp(-2t^2n)$. We refer readers to Chvátal (1979) for more details about the proof of Theorem 2.

Now we prove Theorem 1.

Proof. Taking into consideration both Eqs. (1) and (4), we have

$$\mathbb{P}\left(X \leqslant \alpha K\right) \leqslant \exp\left(-2n\left(p - \frac{\alpha K}{n}\right)^2\right) \leqslant \delta, \tag{5}$$

where $\alpha K = (p-t)n$.

By solving the inequality $\exp\left(-2n\left(p-\frac{\alpha K}{n}\right)^2\right) \leqslant \delta$, we can obtain the sample size that satisfies Eq. (3). □

If the upper bound δ increases or a smaller fraction α is needed, Eq. (3) implies a decrease in the minimum sample size, which seems intuitive since we have less strict requirements. In addition, according to Eq. (3), the minimum sample size decreases when the number of data points belonging to the cluster c, i.e., K, increases with α, N, and δ unchanged. This implies that if there exist several clusters $c_i,\ 1 \leqslant i \leqslant s,\ i, s \in \mathbb{N}$, each of which has K_i data points, it is sufficient to perform a random sampling with size $n_{\min}$ which satisfies Eq. (3) when $K = K_{\min} = \min_{1\leqslant i\leqslant s}(K_i),\ i, s \in \mathbb{N}$. As such, Eq. (3) holds for all clusters with $n \geqslant n_{\min}$, where

$$n_{\min} = \frac{1}{4p_{\min}^2}\left(4p_{\min}\alpha K_{\min} + \log\frac{1}{\delta} + \sqrt{8p_{\min}\alpha K_{\min}\log\frac{1}{\delta} + \left(\log\frac{1}{\delta}\right)^2}\right), \tag{6}$$

with $p_{\min} = \frac{K_{\min}}{N}$.

Moreover, given s clusters, the upper bound for probability that data points from any one of these clusters are fewer than αK is

Table 6
Minimum sample size for each powertrain dataset.

	$K_{\min}$	N	α	δ	s	$n_{\min}$	$n_{\min}/N$ (%)	P
Conventional	7938	61,236	0.3	0.001	4	20,420	**33.3**	0.004
BEV200	55	486	0.3	0.01	3	421	86.6	0.03
EREV PHEV30	215	432	0.3	0.001	2	180	41.7	0.002
EREV PHEV50	213	432	0.3	0.001	2	181	41.9	0.002
Fuel Cell HEV	45	486	0.3	0.02	4	475	97.7	0.08
Par HEV	1314	6102	0.3	0.001	4	2239	**36.7**	0.004
Split HEV	90	432	0.3	0.001	4	279	64.6	0.004

given by following Theorem.

Theorem 3. *Given s clusters, if Eq. (6) holds, the probability that data points from any one of these clusters are fewer than αK is bounded above by $s\delta$.*

Proof. Let A_i denote the event that data points from the i^{th} cluster are fewer than αK_i, where $1 \leqslant i \leqslant s,\ \ i, s \in \mathbb{N}$. Then the probability that data points from any one of these clusters are fewer than αK can be written as $\mathbb{P}(\cup_i A_i)$. Furthermore, we have the following inequality.

$$\mathbb{P}\left(\cup_i A_i\right) \leqslant \sum_{i=1}^{s} \mathbb{P}(A_i)$$

Since Eq. (6) holds, we have

$$\mathbb{P}(A_i) = \mathbb{P}(X_i \leqslant \alpha K_i) \leqslant \delta, \quad \forall\, i \in [1, s],\ \ i \in \mathbb{N}$$

Therefore,

$$\mathbb{P}\left(\cup_i A_i\right) \leqslant \sum_{i=1}^{s} \mathbb{P}(A_i) \leqslant s\delta \quad \square$$

Combined with the results of clustering analysis, we can derive the minimum sample size for each vehicle dataset using Eq. (6) as shown in Table 6, where $K_{\min}$ is the number of data points of the smallest cluster and s the number of clusters in the dataset obtained by clustering analysis. P denotes the upper bound for the probability of missing any one of the clusters. $n_{\min}/N$ is the sampling fraction, i.e., the minimal ratio of sample size to population size. α is set to 0.3 according to our experience.

The minimum sample sizes are most instructive for large vehicle datasets, i.e., conventional and par HEV. Since other vehicle datasets contain only hundreds of data points, the procurement of training sets as well as training processes can be performed with ease. Thus, we can even use their entire inputs for simulation and training. On the other hand, random sample brings more interesting benefit for large datasets. By sampling only 33.3% data points for conventional and 36.7% for par HEV respectively, we are able to obtain a balanced sample almost certainly, which implies that a machine learning model trained on the sample is likely to make reliable prediction. It is also worth pointing out that the minimum sampling fractions for conventional, PHEV and par/split HEV coincide with the sampling fraction given by our random-sampling-based numerosity reduction routine (see Table 4). The result confirms the effectiveness of our sample size analysis.

4.3. Comparing random sampling to stratified sampling

In this section, we compare random sampling to stratified sampling. Stratified sampling refers to a sampling technique that divides the entire population into several non-overlapping homogeneous subpopulations called strata (Lohr, 2009). We then perform independent random sampling for each stratum. Samples from each stratum are assembled to obtain overall population estimates. In the vehicle datasets, vehicles with different classes naturally form strata. By dividing a vehicle dataset into strata and performing random sampling within each stratum, we expect to avoid neglecting subpopulation.

Our experimental results demonstrate that stratified sampling can obtain slightly better prediction accuracy and lower variance in MSE, especially when the sample size is small as shown in Fig. 12. The outputs of BEV 200 refer to vehicle curb weight, acceleration/passing time, electrical energy consumption on UDDS/HWFET/US06, and range on UDDS/HWFET.

We also notice that the larger the sample size, the less difference in accuracy between random sampling and stratified sampling. Intuitively, larger sized samples are more likely to contain sufficient representative data points. In addition, according to our analytically derived upper bound, with high probability, moderate-sized random sampling is able to draw sufficient data points from all clusters. For most of the vehicle datasets, there is no significant difference in accuracy between random sampling and stratified sampling. Therefore, random sampling suffices to reduce simulations in our numerosity reduction routine.

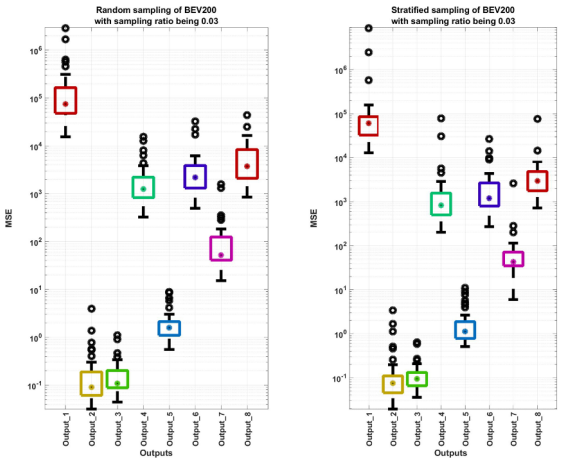

Fig. 12. Comparing random sampling with stratified sampling: we draw a 3% of data points from BEV200 vehicles via random sampling (left) and stratified sampling (right), respectively. Stratified sampling yields slightly better prediction accuracy and lower variance in MSE.

4.4. Summary of sampling strategies

Our theoretical analysis and experimental results confirm that using a moderate-sized random samples can achieve high prediction accuracy. We summarize our sampling strategies as follows.

- We perform random sampling for all vehicle datasets with sample sizes suggested in Table 6 except fuel cell HEV.
- We utilize the entire datasets of fuel cell HEV.
- For small sample sizes such as 3% of the entire data, the stratified sampling technique enables to prevent missing clusters and obtain better prediction accuracy and lower variance in MSE.
- There is no significant difference in accuracy between random sampling and stratified sampling for moderate sample sizes.

5. Future work

Inspired by our current framework, we may extend our work in several directions.

5.1. Anomaly detection for signals

Despite Autonomie's ability to perform reliable vehicle simulation, it is rare yet possible that some simulations are unrealistic even though final outputs such as fuel economy seem normal. For example, due to certain control too complicated to fulfill, excessive shifting events may occur on a specific cycle. By inspecting the intermediate time-based result related to gear control, we are able to discover the anomaly which we cannot otherwise if only the fuel economy result is presented. In order to improve the robustness of simulation in Autonomie, an anomaly detection process for intermediate time-based signals is required to identify events or observations which do not conform to general patterns in a dataset.

5.2. Signal prediction

There is also a desire to achieve analytic continuation on the current discrete space of time series results as we did for simulation outputs such as fuel economy. Such prediction is more challenging than common time series prediction task in that an entire trajectory of signals needs to be predicted instead of predicting future time series based on the current and past series. One possible approach is to represent a discretized signal using a multivariate random variable, which is considered as an output of a neural network. We can then follow the same process as described above to train models and predict signals with respect to new inputs.

Experiments have been carried out on synthetic signals in the form of

$$y = \alpha \cdot \arctan(t) \cdot \sin(\beta t),$$

where t is time, α and β are parameters. By varying α and/or β, we can generate an ensemble of signals as shown in Fig. 13.

Following the same process as explained above, we successfully trained a multilayer feedforward neural network that yields high prediction accuracy (Fig. 14). Simple as the synthetic data are, the experiments can give us insight into this real-world problem.

5.3. Data management and sharing

Another possible extension is related to efficient data management and storage. With our process populating fuel economy

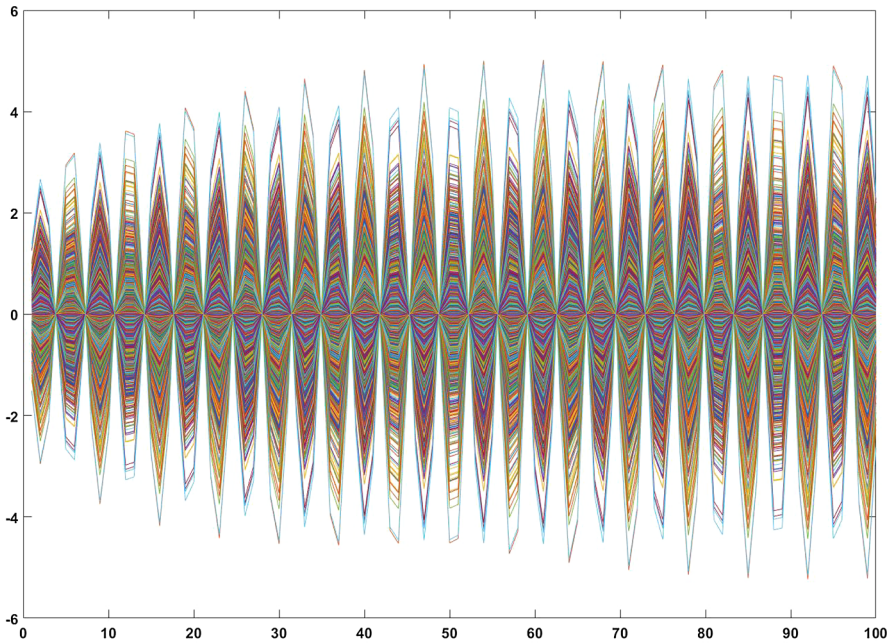

Fig. 13. 1000 signals generated using uniformly distributed random α with β being 10.

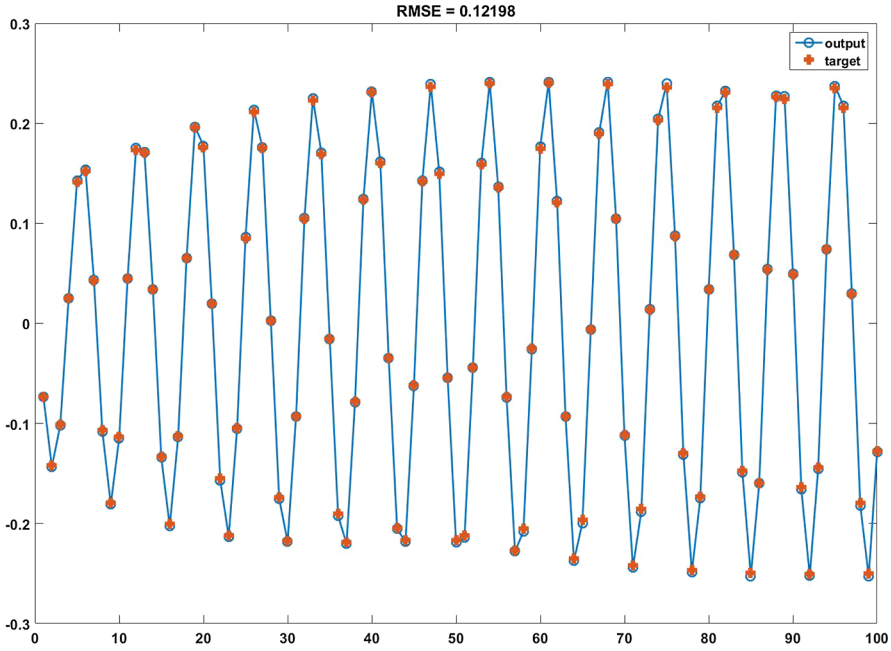

Fig. 14. Prediction of signals with RMSE being 0.12.

prediction, a database that provides large-scale storage and facilitates data management can provide necessary support for our process. It is also possible to construct the database in such a way that data sharing and exchange between users is allowed.

5.4. Real world drive cycles and simulation of cities

The current set of simulated results rely on regulatory City and Highway drive cycles. A next natural step is to account for a richer ensemble of drive cycles in particular make use of real world GPS collected speed profiles. The learning steps will involve ML for eco-routing, impact of energy on people's decision, etc. Simulating entire cities for energy consumption and different fleet composition with multiple vehicle technologies combined with different trips will be greatly facilitated once a drive cycle learning framework is established.

6. Conclusion

This paper proposes an efficient large-scale learning and prediction process via machine learning approaches in order to accomplish analytic continuation on the current discrete space of simulation results. Moreover, we further increase efficiency of simulation and data procurement via our random-sampling-based numerosity reduction routine. We analytically derive a theoretical bound in order to ensure the effectiveness of random sampling. We also compare random sampling to stratified sampling and suggest sampling strategies. A user-friendly machine learning and prediction tool is designed to support the process. Our experimental results confirm that the proposed large-scale learning and prediction process is able to greatly accelerate prediction and analysis of fuel economy.

Acknowledgement

This work was supported by the U.S. Department of Transportation, NHTSA and the Volpe Center. The submitted manuscript has been created by UChicago Argonne, LLC, Operator of Argonne National Laboratory ("Argonne"). Argonne, a U.S. Department of Energy Office of Science laboratory, is operated under Contract No. DE–AC02–06CH11357. The U.S. Government retains for itself, and others acting on its behalf, a paid-up nonexclusive, irrevocable worldwide license in 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.

Appendix A. Tables

See Tables A.7–A.16.

Table A.7
Mapping display names to variable names.

Display name	Variable name
Vehicle Class	Vehicle_Class
Vehicle Powertrain	Vehicle_Powertrain
Engine Type (nb)	Engine_Type
Engine Fuel type	Engine_Fuel_type
Engine Cylinder Configuration	Engine_Cylinder_Configuration
Engine nb. of Cylinders	Engine_nb_of_Cylinders
Engine Displacement: cc	Engine_Displacement
Engine has Cylinder Deac.	Engine_has_Cylinder_Deac
Engine nb. of Banks	Engine_nb_of_Banks
Engine Cam Shaft Configuration	Engine_Cam_Shaft_Configuration
Engine Injection Type	Engine_Injection_Type
Engine Valvetrain Type	Engine_Valvetrain_Type
Engine EGR Type	Engine_EGR_Type
Engine has Turbo	Engine_has_Turbo
Engine Boost Level	Engine_Boost_Level
Engine Max Power: kW	Engine_Max_Power
Max Toque of Pwt: N.m	Max_Toque_of_Pwt
Transmission Type	Transmission_Type
Transmission nb of gears	Transmission_nb_of_gears
Glider Mass Reduction Step	Glider_Mass_Reduction_Step
Glider Mass Reduction Percentage: %	Glider_Mass_Reduction_Percentage
Glider Mass: kg	Glider_Mass
Aerodynamics Reduction Step	Aerodynamics_Reduction_Step
Aerodynamics Reduction Percentage: %	Aerodynamics_Reduction_Percentage
Drag Coefficient	Drag_Coefficient
Rolling Resistance Reduction Step	Rolling_Resistance_Reduction_Step
Rolling Resistance Reduction Percentage: %	Rolling_Resistance_Reduction_Percentage
Rolling Resistance	Rolling_Resistance
Vehicle Test Weight: kg	Vehicle_Test_Weight
Vehicle Curb Weight: kg	Vehicle_Curb_Weight
Gross Vehicle Weight: kg	Gross_Vehicle_Weight
Vehicle Sized	Vehicle_Sized
Vehicle Acceleration Time: s	Vehicle_Acceleration_Time
Vehicle Passing Time: s	Vehicle_Passing_Time
Accessory load: kW	Accessory_load
Motor 1 Max Power: kW	Motor_1_Max_Power
Battery Type	Battery_Type
Battery Pack Power: kW	Battery_Pack_Power
Battery SOC Window: %	Battery_SOC_Window
Battery Total Energy Beginning of Life: W.h	Battery_Total_Energy_Beginning_of_Life

(*continued on next page*)

Table A.7 (*continued*)

Display name	Variable name
Battery Total Energy End of Life: W.h	Battery_Total_Energy_End_of_Life
Fuel Cell Max Power: kW	Fuel_Cell_Max_Power
Unadjusted Fuel Economy UDDS from 2cycle procedure (Gas. Equivalent): mile/gallon	Unadjusted_FE_UDDS
Unadjusted Fuel Economy HWFET from 2cycle procedure (Gas. Equivalent): mile/gallon	Unadjusted_FE_HWFET
Unadjusted PHEV Fuel Economy CD on UDDS (Gas. Equivalent): mile/gallon	Unadjusted_PHEV_FE_CD_UDDS
Unadjusted PHEV Electrical Energy Consumption CD on UDDS: W.h/mile	PHEV_EEC_CD_UDDS
Unadjusted PHEV Fractional EV Range (Rdca) on UDDS: mile	PHEV_Fractional_EV_Range_UDDS
Unadjusted PHEV Fuel Economy CD on HWFET (Gas. Equivalent): mile/gallon	Unadjusted_PHEV_FE_CD_HWFET
Unadjusted PHEV Electrical Energy Consumption CD on HWFET: W.h/mile	PHEV_EEC_CD_HWFET
Unadjusted PHEV Fractional EV Range (Rdca) on HWFET: mile	PHEV_Fractional_EV_Range_HWFET
Unadjusted PHEV Fuel Economy CS on UDDS (Gas. Equivalent): mile/gallon	Unadjusted_PHEV_FE_CS_UDDS
Unadjusted PHEV Fuel Economy CS on HWFET (Gas. Equivalent): mile/gallon	Unadjusted_PHEV_FE_CS_HWFET
Utility-weighted Unadjusted PHEV Fuel Economy on UDDS, (Gas. Equivalent): mile/gallon	Utility_PHEV_FE_UDDS
Utility-weighted Unadjusted PHEV Fuel Economy on HWFET, (Gas. Equivalent): mile/gallon	Utility_PHEV_FE_HWFET
Unadjusted BEV Electrical Energy Consumption on UDDS: W.h/mile	Unadjusted_BEV_EEC_UDDS
Unadjusted BEV Range on UDDS: mile	Unadjusted_BEV_Range_UDDS
Unadjusted BEV Electrical Energy Consumption on HWFET: W.h/mile	Unadjusted_BEV_EEC_HWFET
Unadjusted BEV Range on HWFET: mile	Unadjusted_BEV_Range_HWFET
Unadjusted Fuel Economy US06 (Gas. Equivalent): mile/gallon	Unadjusted_FE_US06
Unadjusted Electric Consumption US06: W.h/mile	Unadjusted_EC_US06

Table A.8
Cross-validation results of BEV200 vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	0.592	0.439	0.728	1920	0.0378
Vehicle_Acceleration_Time	2.51e−05	5.89e−06	0.00498	7.62	0.0653
Vehicle_Passing_Time	0.000864	0.000149	0.0293	7.49	0.391
Unadjusted_BEV_EEC_UDDS	0.296	0.0648	0.541	261	0.207
Unadjusted_BEV_Range_UDDS	0.263	0.0617	0.51	204	0.249
Unadjusted_BEV_EEC_HWFET	0.0745	0.0215	0.27	267	0.101
Unadjusted_BEV_Range_HWFET	0.0608	0.0142	0.245	202	0.121
Unadjusted_EC_US06	0.0142	0.00563	0.117	468	0.025

Table A.9
Cross-validation results of EREV PHEV30 vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	1.13	1.02	0.993	1740	0.0571
Vehicle_Acceleration_Time	8.78e−05	8.15e−05	0.00878	8.23	0.107
Vehicle_Passing_Time	0.0017	0.000547	0.0408	7.99	0.511
PHEV_EEC_CD_UDDS	0.0229	0.0114	0.148	203	0.073
PHEV_Fractional_EV_Range_UDDS	0.000379	0.000156	0.0191	30.8	0.0622
PHEV_EEC_CD_HWFET	0.0832	0.0371	0.282	238	0.119
PHEV_Fractional_EV_Range_HWFET	0.00108	0.000522	0.032	26.3	0.122
Unadjusted_PHEV_FE_CS_UDDS	0.0272	0.0166	0.16	55.1	0.289
Unadjusted_PHEV_FE_CS_HWFET	0.0037	0.00156	0.0595	54.3	0.11
Utility_PHEV_FE_UDDS	0.151	0.0697	0.381	120	0.317
Utility_PHEV_FE_HWFET	0.146	0.0655	0.371	106	0.35
Unadjusted_EC_US06	0.0116	0.00576	0.105	345	0.0305

Table A.10
Cross-validation results of EREV PHEV50 vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	0.762	0.315	0.858	1870	0.046
Vehicle_Acceleration_Time	6.1e−05	2.34e−05	0.00768	7.74	0.0993
Vehicle_Passing_Time	0.0019	0.000365	0.0434	7.43	0.584
PHEV_EEC_CD_UDDS	0.0922	0.0259	0.301	207	0.145
PHEV_Fractional_EV_Range_UDDS	0.0048	0.000875	0.069	51	0.135
PHEV_EEC_CD_HWFET	0.51	0.167	0.705	241	0.293
PHEV_Fractional_EV_Range_HWFET	0.0111	0.00442	0.103	44.1	0.234
Unadjusted_PHEV_FE_CS_UDDS	0.267	0.117	0.505	52.5	0.962
Unadjusted_PHEV_FE_CS_HWFET	0.00822	0.00215	0.09	52.7	0.171
Utility_PHEV_FE_UDDS	2.69	1.59	1.58	173	0.914
Utility_PHEV_FE_HWFET	0.829	0.373	0.891	154	0.58
Unadjusted_EC_US06	0.0188	0.0128	0.131	352	0.0373

Table A.11
Cross-validation results of Fuel Cell HEV vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	15.4	24.2	3.28	1700	0.193
Vehicle_Acceleration_Time	0.000184	0.000176	0.0127	8.86	0.143
Vehicle_Passing_Time	0.00365	0.00397	0.0555	7.07	0.784
Unadjusted_FE_UDDS	0.00557	0.00691	0.0655	73.3	0.0893
Unadjusted_FE_HWFET	0.00403	0.00571	0.0531	78.6	0.0675
Unadjusted_FE_US06	0.00892	0.00509	0.0916	−19.9	−0.461

Table A.12
Cross-validation results of micro hybrid vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	0.0327	0.0123	0.178	1510	0.0118
Vehicle_Acceleration_Time	0.000746	0.000327	0.0268	8.6	0.312
Vehicle_Passing_Time	0.00119	3.87e−05	0.0345	6.58	0.525
Unadjusted_FE_UDDS	0.00874	0.00131	0.0932	36	0.259
Unadjusted_FE_HWFET	0.00979	0.00132	0.0987	48.6	0.203
Unadjusted_FE_US06	0.0586	0.0198	0.239	29.7	0.804

Table A.13
Cross-validation results of mild hybrid BISG vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	0.0229	0.00597	0.15	1520	0.00991
Vehicle_Acceleration_Time	0.000432	0.000157	0.0205	8.66	0.237
Vehicle_Passing_Time	0.00109	4.48e−05	0.0331	6.74	0.491
Unadjusted_FE_UDDS	0.00543	0.00126	0.0733	38.6	0.19
Unadjusted_FE_HWFET	0.00733	0.00219	0.0848	50.2	0.169
Unadjusted_FE_US06	0.0149	0.00802	0.118	30.6	0.386

Table A.14
Cross-validation results of mild hybrid CISG vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	0.0256	0.0143	0.155	1520	0.0102
Vehicle_Acceleration_Time	9.76e−05	1.19e−05	0.00986	8.65	0.114
Vehicle_Passing_Time	0.00109	5.42e−05	0.033	6.88	0.48
Unadjusted_FE_UDDS	0.285	0.0224	0.534	41.3	1.29
Unadjusted_FE_HWFET	0.00405	0.001	0.0632	51.4	0.123
Unadjusted_FE_US06	0.0197	0.00985	0.136	31.6	0.43

Table A.15
Cross-validation results of par HEV vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	15.4	0.0801	0.571	1540	0.0371
Vehicle_Acceleration_Time	0.000184	1.66e−06	0.00493	8.85	0.0557
Vehicle_Passing_Time	0.00365	6.05e−05	0.0354	6.72	0.527
Unadjusted_FE_UDDS	0.00557	0.00112	0.0994	55.9	0.178
Unadjusted_FE_HWFET	0.00403	0.000207	0.0383	56.2	0.0681
Unadjusted_FE_US06	0.00892	0.0207	0.107	36.4	0.295

Table A.16
Cross-validation results of split HEV vehicles.

	$\overline{\text{MSE}}$	σ_{MSE}	$\overline{\text{RMSE}}$	$\overline{Y}$	CV($\overline{\text{RMSE}}$) (%)
Vehicle_Curb_Weight	0.986	0.69	0.95	1570	0.0607
Vehicle_Acceleration_Time	0.00219	0.000556	0.0465	9.03	0.515
Vehicle_Passing_Time	0.00325	0.000561	0.0568	5.98	0.949
Unadjusted_FE_UDDS	0.00856	0.00208	0.0918	57.4	0.16
Unadjusted_FE_HWFET	0.00138	0.00137	0.0336	56.5	0.0594
Unadjusted_FE_US06	0.00407	0.0013	0.0631	39	0.162

Appendix B. Notation

Y	target
$\widehat{Y}$	predicted output
$\overline{Y}$	mean of target Y
MSE	mean square error, $\frac{1}{n}\sum_{i=1}^{n}(\widehat{Y}_i - Y_i)^2$
$\overline{\text{MSE}}$	average MSE
σ_{MSE}	standard deviation of MSE
RMSE	root-mean-square error, $\sqrt{\text{MSE}}$
$\overline{\text{RMSE}}$	average RMSE, $\sqrt{\overline{\text{MSE}}}$
CV($\overline{\text{RMSE}}$)	coefficient of variation of the $\overline{\text{RMSE}}$, $\frac{\overline{\text{RMSE}}}{\overline{Y}}$
$D_{\text{KL}}(a\|\|b)$	Kullback-Leiblier divergence, $a\log\frac{a}{b} + \left(1-a\right)\log\frac{1-a}{1-b}$
c	an arbitrary cluster
N	total number of data points in the dataset
K	number of data points in the cluster c of interest
n	number of points in the sample
k	number of points belonging to c in the n sample points
p	fraction of points in the cluster c, $\frac{K}{N}$
δ	upper bound of probability, $0 \leqslant \delta \leqslant 1$
α	fraction of sample points belonging to c, $0 \leqslant \alpha \leqslant 1$
s	number of clusters in the dataset
P	upper bound for the probability of missing any one of the clusters
BEVx	battery-powered electric vehicle with x miles of all-electric range
CD	charge depleting
CS	charge sustaining
EREV	extended-range electric vehicle
HEV	hybrid electric vehicle
HWFET	Highway Federal Emissions Test
PHEVx	plug-in hybrid electric vehicle with x miles of all-electric range
SOC	state of charge
SUV	sport utility vehicle
UDDS	Urban Dynamometer Driving Schedule

References

Abdi, H., Williams, L.J., 2010. Principal component analysis. Wiley Interdiscip. Rev.: Comput. Stat. 2, 433–459.
Autonomie, 2014. http://www.autonomie.net/.
Barnett, V., Lewis, T., 1994. Outliers in Statistical Data. third ed. John Wiley & Sons.
Bishop, C.M., 2006. Pattern Recognition and Machine Learning (Information Science and Statistics). Springer-Verlag, Berlin, Heidelberg.

Chryssolouris, G., Lee, M., Ramsey, A., 1996. Confidence interval prediction for neural network models. IEEE Trans. Neural Networks 7, 229–232.
Chvátal, V., 1979. The tail of the hypergeometric distribution. Discr. Math. 25, 285–287.
Efron, B., 1982. The Jackknife, the Bootstrap and Other Resampling Plans. SIAM.
Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM 24, 381–395.
Garavaglia, S., Sharma, A., 1998. A smart guide to dummy variables: four applications and a macro. In: Proceedings of the Northest SAS Users Group Conference.
Han, J., Pei, J., Kamber, M., 2006. Data Mining, second ed. Morgan Kaufmann, Southeast Asia, pp. 99–120 (Chapter 3).
Hartigan, J.A., Wong, M.A., 1979. A K-means clustering algorithm. Appl. Stat. 28, 100–108.
Hornik, K., Stinchcombe, M., White, H., 1989. Multilayer feedforward networks are universal approximators. Neural Netw. 2, 359–366.
Kaufman, L., Rousseeuw, P.J., 1987. Clustering by means of medoids. Statis. Data Anal. L1 Norm 405–416.
Kohavi, R., Foster, P., 1998. Glossary of terms. Mach. Learn. 30, 271–274.
Li, S.E., Guo, Q., Xin, L., Cheng, B., Li, K., 2017. Fuel-saving servo-loop control for an adaptive cruise control system of road vehicles with step-gear transmission. IEEE Trans. Veh. Technol. 66, 2033–2043. https://doi.org/10.1109/TVT.2016.2574740.
Lohr, S.L., 2009. Sampling: Design and Analysis, second ed. Cengage Learning, Inc.
Moawad, A., Balaprakash, P., Rousseau, A., Wild, S., 2016. Novel large scale simulation process to support DOT's CAFE modeling system. Int. J. Automot. Technol. (IJAT) 17, 1067–1077.
Moawad, A., Rousseau, A., 2014. Light-duty Vehicle Fuel Consumption Displacement Potential up to 2045. Argonne National Laboratory.
Paass, G., 1993. Assessing and improving neural network predictions by the bootstrap algorithm. In: In: Hanson, S.J., Cowan, J.D., Giles, C.L. (Eds.), Advances in Neural Information Processing Systems, vol. 5. Morgan-Kaufmann, pp. 196–203.
Svozil, D., Kvasnicka, V., Pospichal, J., 1997. Introduction to multi-layer feed-forward neural networks. Chemometr. Intell. Lab. Syst. 39, 43–62.
Tibshirani, R., 1996. A comparison of some error estimates for neural network models. Neural Comput. 8, 152–163.
Xu, S., Li, S.E., Cheng, B., Li, K., 2017. Instantaneous feedback control for a fuel-prioritized vehicle cruising system on highways with a varying slope. IEEE Trans. Intell. Transport. Syst. 18, 1210–1220. https://doi.org/10.1109/TITS.2016.2600641.
Xu, S., Li, S.E., Peng, H., Cheng, B., Zhang, X., Pan, Z., 2016. Fuel-saving cruising strategies for parallel hevs. IEEE Trans. Veh. Technol. 65, 4676–4686. https://doi.org/10.1109/TVT.2015.2490101.