TRR
JOURNAL OF THE TRANSPORTATION RESEARCH BOARD

Article

Transportation Research Record
1–12
© National Academy of Sciences:
Transportation Research Board 2018
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/0361198118796971
journals.sagepub.com/home/trr
SAGE

Comparison of Four Types of Artificial Neural Network and a Multinomial Logit Model for Travel Mode Choice Modeling

Dongwoo Lee[1], Sybil Derrible[1], and Francisco Camara Pereira[2]

Abstract

Discrete choice modeling is a fundamental part of travel demand forecasting. To date, this field has been dominated by parametric approaches (e.g., logit models), but non-parametric approaches such as artificial neural networks (ANNs) possess much potential since choice problems can be assimilated to pattern recognition problems. In particular, ANN models are easily applicable with their higher capability to identify nonlinear relationships between inputs and designated outputs to predict choice behaviors. This article investigates the capability of four types of ANN model and compares their prediction performance with a conventional multinomial logit model (MNL) for mode choice problems. The four ANNs are: backpropagation neural networks (BPNNs), radial basis function networks (RBFNs), probabilistic neural networks (PNNs), and clustered probabilistic neural networks (CPNNs). To compare the modeling techniques, we present the algorithmic differences of each ANN technique, and we assess their prediction accuracy with a 10-fold cross-validation method. Furthermore, we assess the contribution of explanatory variables by conducting sensitivity analyses on significant variables. The results show that ANN models outperform MNL, with prediction accuracies around 80% compared with 70% for MNL. Moreover, PNN performs best out of all ANNs, especially to predict underrepresented modes.

In travel mode choice modeling, and based on a set of given attributes, the objective is to model the choice processes and behaviors of travelers faced with several travel mode alternatives. Being able to model mode choice and predict future trends is critical to transportation planners and policy makers. Travel behaviors are typically examined by using statistical survey techniques such as reveal preference (RP) and stated preference (SP), in which respondents are asked to present their choices and preferable scenarios, respectively (*1*). The surveys yield discrete choice data that can then be used to calibrate travel mode choice models that are fundamental components of disaggregate travel demand modeling approaches, for example, activity-based modeling (ABM).

As the main parametric modeling approach, random utility modeling has been the dominant technique used in the literature since the 1980s to investigate parametric relationships between mode choice and its possible determinants. In particular, logit models—one of the random utility models—gained popularity in the travel mode choice realm because they are based on simple mathematical formulations while accounting for unobserved variables (i.e., stochasticity). Logit models (e.g., multinomial logit [MNL]), however, assume that each choice is independent and identically distributed (IID), which can lead to biased estimations and misleading predictions. Moreover, interactions between the explanatory variables (e.g., the nonlinearity of attributes) are often neglected since it is difficult to represent them in the utility function (e.g., conditional and threshold effects). One way to take this issue into account in the linear utility function is by introducing additional variables (e.g., polynomial and interaction terms) either through considering all possible combinations of explanatory variables or by measuring empirical relationships (*2*). Nonetheless, it is practically impossible to identify all the interactions between all the variables as well as to identify the necessary variables. MNL tries to overcome this issue by using domain knowledge as much as possible. Although the flexible random utility model (RUM) methods such as

[1] Department of Civil and Materials Engineering, University of Illinois at Chicago, Chicago, IL
[2] DTU Management Engineering, Technical University of Denmark, Kongens Lyngby, Denmark

Corresponding Author:
Address correspondence to Dongwoo Lee: dlee226@uic.edu

mixed logit (*3*–*5*) can have better modeling performance than MNL by relaxing IID assumptions, predetermined structures and linear characteristics of underlying functions still make it difficult to capture or infer high degrees of nonlinearity in a dataset (*3*, *6*, *7*).

In contrast, algorithmic non-parametric approaches have proven to be incredibly powerful to study urban systems (*8*–*15*). Specifically, artificial neural networks (ANNs) present higher adaptability in identifying nonlinear interactions (*16*, *17*). Contrary to the logit model that deals with the nonlinearity issue by reducing the "complexity" of the dataset, ANNs can capture nonlinear properties more easily through additional units (e.g., hidden layers) without assuming predetermined functional forms (*1*, *18*–*20*). Nonetheless, a commonly acknowledged disadvantage of ANNs is their lack of interpretability, and they are often associated with black boxes. Another disadvantage is their relative inability to use previously acquired knowledge (i.e., domain knowledge). Despite these challenges, ANNs tend to outperform logit models (from the multinomial to nested logit models [*20*–*22*]). Among different types of ANN models, several studies have used backpropagation neural networks (BPNNs) to model mode choice (*19*, *21*, *23*). BPNNs tend to achieve high prediction accuracy and are easy to apply. Nonetheless, many different types of ANN models exist, often superior to BPNNs, combining both a strong statistical background with machine learning features. Probabilistic neural networks (PNNs), for example, are derived by incorporating statistical features (e.g., Bayesian decision rule and kernel density estimation [KDE]) into the structure of the neural network. In addition, additional treatments on the dataset or network structure are applied to obtain more efficient and reliable models.

In this article, we investigate the feasibility and capability of four ANNs to model discrete mode choice behaviors and compare their prediction performance with a MNL. Specifically, there have been a limited number of articles that conduct comprehensive analyses of different types of ANN algorithms to model discrete choice in the field of transportation. In this study, we focus on four types of ANNs: BPNNs, radial basis function networks (RBFN), PNNs, and clustered probabilistic neural networks (CPNNs). As a typical travel survey, the Chicago Metropolitan Area for Planning (CMAP) Travel Tracker Survey dataset collected from 2007 to 2008 is used. This study consists of six sections. After this introduction, we present the five methods for discrete choice modeling. We then explain how the data were prepared. Thereafter we provide the model specifications for MNL and ANN models and follow this with the results and a discussion, before the conclusion.

Methods for Discrete Choice Modeling

Random Utility Model: Logit Models

The logit model is the most popular type of random utility model derived from consumer economics theory, and it was initially developed by McFadden (*3*, *24*). In utility maximization behavior, an individual i makes a decision to select one choice among discrete alternatives, by evaluating their associated attributes $\boldsymbol{X}$. The individual i chooses the alternative m that provides the largest utility:

$$U_{im} > U_{ik} \quad \forall m \neq k \tag{1}$$

In reality, researchers do not observe the complete utility of the individual. Thus, the utility can be classified into two parts: an observed utility V_{im} and an unobserved utility ε_{im}. The observed utility generally contains two sets of attributes: 1) covariates associated with both the individual and the alternative X_{im}; and 2) decision-maker characteristics, S_i (*3*). The observed (stated) utility (V) is a value determined from a linear combination of the attributes used, which captures the attractiveness of an alternative bounded to the given model specification as follows:

$$V_{im} = V(X_{im}, S_i) \tag{2}$$

In contrast, the unobserved utility ε_{im} cannot be observed by researchers. This unobserved part mainly results from the specification of the observed utility V_{im}. In practice, it is impossible for statistical approaches to include all possible attributes. Therefore, researchers treat the unobserved terms as a stochastic element. Specifically, the logit model is derived by assuming that each unobserved term, ε_{lm}, is IID extreme values, i.e., Gumbel and type 1 extreme values. By combining two utilities, we can get the probability of individual i choosing alternative j by solving the mathematical formulation:

$$P_{ij} = \frac{e^{\beta' X_{im}}}{\sum_{k=1}^{M} e^{\beta' X_{ik}}} \tag{3}$$

where $\boldsymbol{X}_{ik}$ is the vector of observed explanatory variables to choose a given alternative, and β' is parameters for the observed utility. For more technical details about logit models, see (*3*).

ANNs

ANNs essentially emulate decision-making processes occurring in the brain. As opposed to logit models that assume linearity in estimation, ANNs can algorithmically construct models that are based on nonlinear relationships between determinants. Figure 1*b* shows a

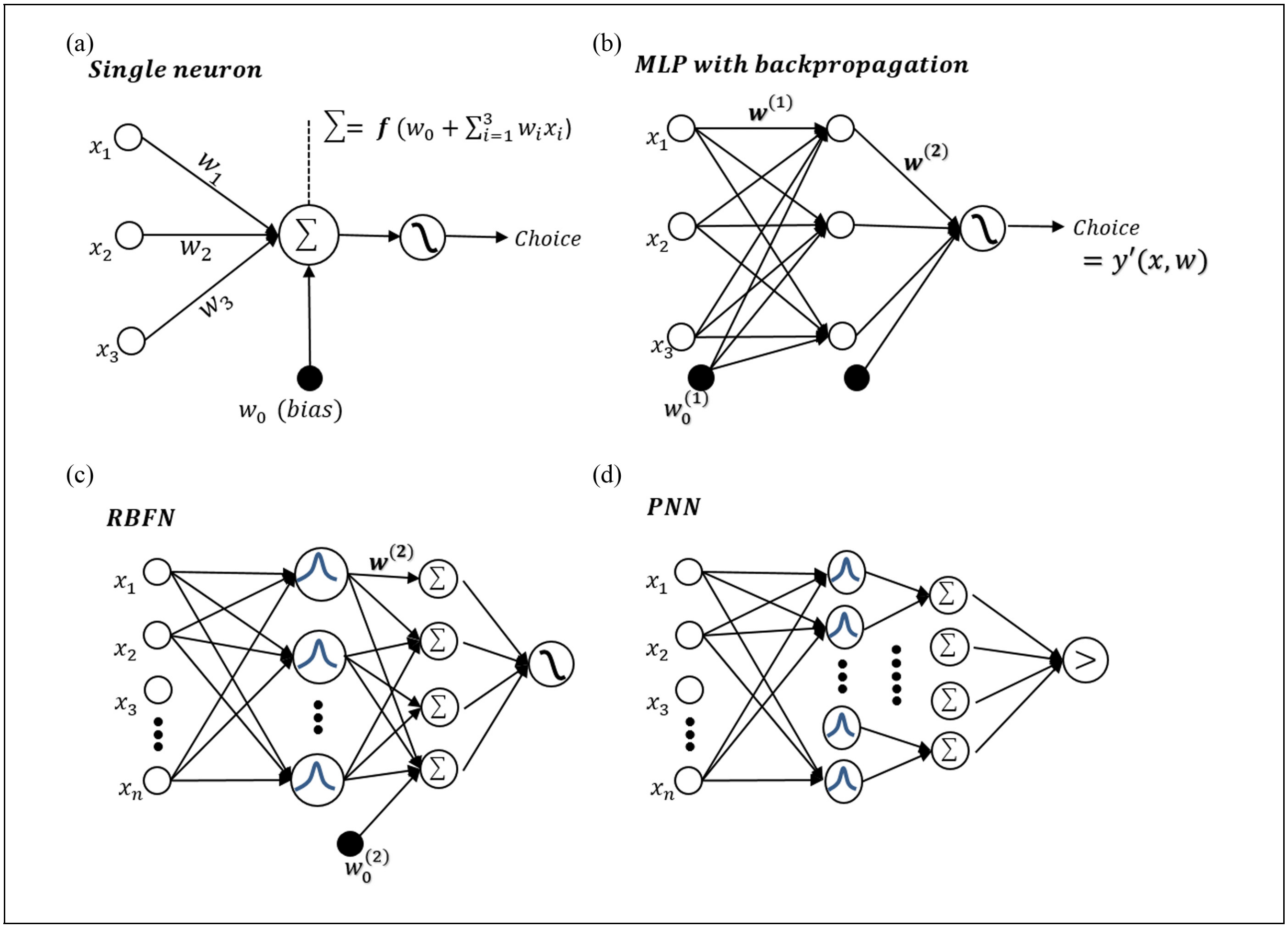

Figure 1. Structure of an ANN: (*a*) single neuron; (*b*) MLP with backpropagation; (*c*) RBFNN; (*d*) PNN.

multi-layer perceptron MLP that has been widely adapted in most neural network models.

In Figure 1*a*, the process of predicting output using a single neuron (i.e., a basis function) is similar to the logit model, since it is based on a linear function. ANNs, however, are constructed by combining multiple neurons to identify nonlinear relationships between a choice and its associated explanatory variables.

Several types of neural networks exist, and the differences between them mainly come from the nature of the basis function for each model. In the following subsections, we detail four different types of neural networks and their structural and mathematical differences. Table 1 contains the main characteristics, advantages, and disadvantages of the four ANN models.

BPNN. BPNN is the most popular and simplest algorithm of MLP. The backpropagation process usually minimizes the error function (Equation 4) of the MLP by changing the value of the weights using a gradient descent method (GDM):

$$E = \sum_{i=1}^{N} \|y_i' - y_i\|^2 \tag{4}$$

where E is the error, i is the number of training samples, y' is the predicted values through the training, and y is the actual values from the training dataset. Each neuron follows the same activation process as shown for a single neuron. Typical activation functions include the step, sigmoid, tanh, rectified linear unit (ReLU), and the identity functions (*25*).

The hidden neurons are "activated" by receiving the sum products of input vectors $\boldsymbol{x}$ and their associated weights $\boldsymbol{w}$. The output layer also obtains information from the hidden layer in the same manner. The final output, y' is:

$$y'(\boldsymbol{x}, \boldsymbol{w}) = \sigma(\sum_{j=1}^{3} w_j^{(2)} \cdot h(\sum_{i=1}^{3} x_i w_i^{(1)} + w_0^{(1)}) + w_0^{(2)}) \tag{5}$$

where $\boldsymbol{x}$ is input vector, $\boldsymbol{w}$ is the vector of associated weights, $w^{(1)}$ are the weights between the input layer and the hidden layer, $w^{(2)}$ is the weights between the hidden

Table 1. Comparison of Four Types of Neural Networks

	BPNN	RBFN	PNN	CPNN
Application scope	Classification/regression	Classification/ regression	Classification	Classification
Classification process	Minimize sum squared errors by updating weights	RBF and BPNN process	Parzen PDF classifier (KDE and Bayesian decision rule)	Use clustering methods and PNN process
Advantages	• Simple application to predict the patterns • Does not require any statistical features in the learning process • Easy to identify the magnitude of attributes based on weights (relative importance) • A variety of applications are available	• Simpler format of Gaussian function enables faster learning process than other Gaussian models • Radial basis function nodes can be substituted with different functional forms • Performs relatively well in both smaller and larger dataset	• Simple architecture (no backpropagation) • More ways to manage the algorithm by determining the shape of bell curve, specifically width (σ) • (more specific than RBFN) • Relatively good accuracy in classification problems • Insensitive to the noise points	• Smaller network size than ordinary PNN • Can avoid saturation of Parzen window that leads to misclassification • May be more applicable because it provides knowledge of relative importance between explanatory variables • Faster training time
Disadvantages	• Easily get stuck in local minima, resulting in suboptimal solution • Like a black box (not sure how to estimate the model) • Need sufficient observations • Prone to overfitting	• Difficult to determine the σ values • Constructing networks architecture is complicated. • Long training time	• More computationally expensive than BPNN (pre-stored pattern neurons) • Saturated Gaussian function can lead some misclassification	• May not provide higher prediction accuracy than PNN for discrete choice data • Varied by number of clusters determined in K-means clustering

layer and the output layer, and the $h()$ and $\sigma()$ functions are the activation functions (see Figure 1*b*).

The drawbacks of the backpropagation process notably come from the GDM that does not guarantee a global minimum is reached when local minima exist. Thus, the results usually vary each time the BPNN is trained. Second, no "optimal" rule exists to set the structural parameters of the BPNN such as the number of hidden layers and their associated number of neurons; although it is generally believed that a larger number of neurons per hidden layer increases the accuracy of the model estimation up to a point (*25*). Nonetheless, in general, one or a few hidden layers are sufficient. Despite these limitations, it is worth mentioning that the BPNN is still the most dominant type of ANN used because it is simple to apply and it ensures relatively high accuracy.

RBFNs. A RBFN is also a feedforward network with three layers: an input, a radial basis function (RBF) hidden, and an output layer (see Figure 1*c*). In contrast to the BPNN, the RBFN is formulated by a linear combination of Gaussian basis functions $g(x)$ as the outputs of the hidden layer:

$$Output\ Score = \sum wh(g(x)) \tag{6}$$

where $h()$ is an activation function, w is a weight between a hidden neuron and an output neuron, and *Output score* is an output for each given class. The basis function $g(x)$ is conceptually obtained by calculating the distance between two vectors based on the Gaussian function whose outputs are inversely proportional to the distance from the mean:

$$g(x) = \frac{1}{\sigma(2\pi)^{\frac{1}{2}}}\exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{c}_i\|^2}{2\sigma^2}\right) = \exp\left(-\beta\|\boldsymbol{x}-\boldsymbol{c}_i\|^2\right) \tag{7}$$

where $\boldsymbol{x}$ is an input vector with n variables (vector to be classified) and $\boldsymbol{c}$ is a "prototype" vector. To be specific, this equation represents the function of the Euclidean distance between the new input vector and the prototype vector. Essentially, the term β determines the width of the Gaussian curve, which controls the decision boundaries. The prototype vector, $\boldsymbol{c}_i$, is pre-stored in each hidden neuron, and it is a preselected data point from the training set, whether randomly or not (discussed later).

Each hidden neuron compares the input vector to its prototype vector to measure how "similar" they are.

The neurons in the output layer are the sum (linear combination) of the weighted hidden layer neurons (i.e., output score), and the final output, y' is determined by activation functions such as sigmoid function:

$$y' = f(\sum_{i=1}^{N} w_i \cdot \exp(-\beta \|\boldsymbol{x} - \boldsymbol{c}_i\|^2)) \tag{8}$$

To calibrate the weights, the GDM is also used. By using the outputs (i.e., activations) from each RBF hidden neuron as inputs, the gradient descent process is separately conducted for each output class. Although it tends to be superior to the BPNN network structure, similar to the BPNN, there is no guarantee that a global minimum in found when optimizing the weights in a RBFN.

The performance of an RBFN depends on the number of neurons in the RBFN hidden layer and their prototype vectors. The simplest way to determine the k prototypes is to select them randomly. While it can improve the accuracy of the model, using more prototypes increases the computational costs of the algorithm, and it may generate overfitting issues as well. Another approach to select the prototypes is first to use a clustering algorithm (discussed later).

PNNs. PNNs add a Bayesian decision rule into the framework of RBFNs (see Figure 1*d*). Technically, a PNN is derived from Bayes–Parzen classifiers (*26*), which include Bayesian classification and classical estimators for probabilistic density function (PDF) (*27*, *28*). Essentially, Parzen's method (*29*) is first used to estimate the PDF of each class (i.e., travel modes) from the provided training samples based on a certain type of kernel. These estimated PDF can then go through a Bayesian decision rule to find a final decision.

For any classification problem, a sample vector $\boldsymbol{x}$ is taken from a collection of samples belonging to several classes. If we assume that the prior probability that a sample belongs to class i is h_i, the loss associated with misclassifying that sample is c_i, and the PDF for each of the populations $f_k(x)$ is known. The Bayesian decision rule for classifying an unknown sample into the ith class can be applied as:

$$h_i c_i f_i(x) > h_j c_j f_j(x) \quad \forall j \neq i \tag{9}$$

The PDF for class $i, f_i(x)$, represents the concentration of class i samples around the unknown sample. That is, the Bayesian decision rule favors a class that has high density adjacent to the unknown sample, or if the cost of misclassification or prior probability is high. However, in the Bayesian decision, the PDF for each class is usually known. Therefore, the Parzen's method (*29*) is usually used to estimate the PDF. The univariate KDE was developed by Parzen and it was then extended to the multivariate case by Cacoullos (*30*). The multivariate KDE for n random (independent) variables can be defined as:

$$\varnothing_i(x_n) = \frac{1}{N\lambda} \sum_{k=1}^{N_i} W\left(\frac{x - x_{ik}}{\lambda}\right) \tag{10}$$

where $\boldsymbol{x}_n$ is the vector of independent variables with $n$numbers, and x_k is kth training vector choosing class i; N_i is the total number of observations in the training dataset for class i; λ represents all kernel width for number of variables (n), $\lambda = [\sigma_1, \sigma_2, \cdots, \sigma_n]$; σ is a smoothing parameter representing kernel width (standard deviation; σ is important to obtain an optimal PNN); W is the weight function, and the most popular type of weight function is Gaussian Kernel, which is mainly derived from the concept of Euclidean distance as:

$$W\left({}^{x - x_{ik}}/_{\lambda}\right) = e^{\left(-\|x - x_{ik}\|^2/2\sigma^2\right)} \tag{11}$$

By employing a Gaussian weight function, the PDF for choosing class i in the given multivariate input vector $\boldsymbol{x}$ can be expressed as follows:

$$\varnothing_\mathrm{i}(1\boldsymbol{x}) = \frac{1}{N_i (2\pi)^{\frac{n}{2}} \sigma^n} \sum_{k=1}^{N_\mathrm{i}} \exp\left(-\|\boldsymbol{x} - x_{ik}\|^2 \Big/ 2\sigma^2\right) \tag{12}$$

The final classification decision for the input vector $\boldsymbol{x}$ can be obtained by applying Bayes decision rule as follows:

$$C(x) = arg \max_{i=}(\varnothing_\mathrm{i}(\boldsymbol{x})) \tag{13}$$

CPNN. An ANN using Gaussian KDE such as the PNN may encounter issues related to the size of the network. Specifically, without any pretreatment of the input dataset, the number of pattern neurons containing the prototype vectors is determined either by the number of training samples or by random selection during the training. Since each prototype vector plays a role in KDE by locating the center of the Gaussian distribution for measuring Euclidean distance, the number of pattern neurons can affect the model performance. As the number of pattern neurons increases, the structure of the PNN will become more complicated and more computationally expensive. In previous studies, two feasible methods have been applied to the conventional PNN: 1) K-means clustering to determine the number of pattern neurons; and 2) self-adaptive learning for smoothing parameter (σ) (*31–33*). Here, this study employs the K-means clustering method. Although determining σ is directly related to the decision boundaries, this study simply assigns a certain value according to the knowledge of the dataset and values selected in other studies.

As an unsupervised learning technique, *K*-means clustering has been widely used to classify the data set into the number of clusters (*34*). Then, the centroid of a cluster (i.e., the mean value of the observations within the cluster) is defined by minimizing the distance between the observations belonging to a cluster and the center of the cluster (for details about *K*-means, see [*34*]). The centroids represent the centers of clusters in the Euclidean space, and they become new prototype vectors to measure a Euclidean distance in the pattern neurons. The *K*-means clustering process is separately applied to each choice alternatives.

Data

The data used to test the four models comes from the CMAP Travel Tracker Survey that was collected in 2007 and 2008. The RP survey includes travel diary information as well as detailed individual and household sociodemographic information, for 10,500 households. The original database consists of four interconnected datasets containing household, person, place, and vehicle information. The first three datasets are consolidated to make the dataset.

To merge the datasets, the ESRI ArcGIS package is used to identify interrelated geographical and travel information such as the location of the origins (O) and the destinations (D), walk accessibility, and the actual distance between Os and Ds. Since our focus is on the comparison of the five models, we only look at home-based trips. Other trips, such as the access and egress trips, transfer trips, and other nonhome-based trips are excluded from the dataset used for this study.

In addition, the alternatives for the home-based trip are classified into four classes: walk, bike, transit (CTA bus and train), and auto. The original database contains only 2–3% of transit observations, which is an issue since the proportion of transit observations in the training and testing datasets can be significantly different (e.g., one of the two could have 0 transit observations). To be able to fairly compare the model accuracies, we select to instead use a sub-sample of the original dataset to include more transit observations. Nonetheless, to assess the performance of each model to select poorly represented modes, we left the bike observations intact, which only account for about 4% of all observations. In the end, the dataset used includes 4,764 observations. It includes two sets of attributes as independent variables: (1) individual/household socio-demographic attributes; and (2) travel attributes (e.g., travel time, cost, accessibility) (see Table 2).

Model Specification

Normalization

Discrete choice data contain a variety set of variables with varying scales and ranges. The different numerical

Table 2. Description of Statistics

Variable	Description	Mean	Std
HHSIZ	Number of household members	2.80	1.43
HHWK	Number of workers in household	1.49	0.92
HHSTU	Number of students in household	0.87	1.14
FEMALE	1: if a traveler is female, 0: otherwise	0.52	0.50
EMPLY	Employment status for traveler	1.59	1.05
EDUCA[a]	Education level	3.77	1.91
STUDE	Student grade	2.64	0.74
HHVEH[a]	Number of vehicles in household	1.66	1.06
BIKES[a]	Number of bikes in household	1.48	1.76
CAPINCOME[a]	Capita income of household	0.75	0.37
WalkTT	Travel time for walk mode (hour)	4.76	3.92
BikeTT	Travel time for bike mode (hour)	1.32	1.53
AutoTT	Travel time for auto mode (hour)	0.88	0.91
TransitTT	Travel time for CTA mode (hour)	0.90	1.30
AUTO_COST	Total trip cost for auto (gas, toll, parking / $)	0.90	1.11
Transit_COST	Total trip cost for CTA ($)	0.96	1.12
AGE[a]	Traveler's age	44.02	21.20
ACT_DUR	Actual activity duration (hours)	3.48	3.58
WALK_ACC.	1: if the walking accessibility is within 0.30 miles, 0: otherwise	0.08	0.28

Note: Std. = Standard deviation.

[a]Employed with either the type of dummy or nominal or categorical.

Table 3. Multinomial Logit Model Estimation for Mode Choice

Variable	Coefficient	*t*-stat
Alternative specific constant (auto is base)		
Walk	2.442	3.48
Bike	–2.033	–11.41
Transit	–1.097	–3.85
Travel time × auto	–0.92	–2.06
Travel time × transit	–1.51	–3.71
Travel time × bike	–0.05	13.87
Auto operating cost × auto	–1.25	–12.931
Transit fare × transit	–2.229	–16.71
Walk accessibility × transit	–0.755	–8.12
Walk accessibility × bike	0.424	2.74
Bikes in HH × bike	1.794	13.18
Number of vehicles in HH × walk	–0.941	–2.08
Number of vehicles in HH × transit	–1.131	–1.98
Age over 75 × walk	–0.358	–4.91
Age over 75 × bike	–1.224	–11.94
Loglikelihood at constant: −4822.92		
Loglikelihood at final convergence: −4121.61		

properties among variables may result in estimation biases. Specifically, the PDF for the Gaussian kernel-based ANN (i.e., RBFN and PNN) cannot be estimated without any pretreatment of the input dataset. Moreover, it is preferable to normalize the input data when the sigmoid function is used in BPNN. Therefore, we normalize all values of attributes in the input dataset before training (estimating) the neural networks specifically; the non-normalized data was used for the MNL as is common in practice. The max-min normalization allows all attributes to be located ranging from 0– 1:

$$x' = \frac{x - \min A}{\max A - \min A} \tag{14}$$

where x' is the new value of the attribute, x is the original value of the attribute; A is the set of all values for a variable from entire data set (training and testing), and $\min A$ and $\max A$ represent the minimum and maximum value of a variable respectively.

Cross-Validation

Cross-validation is a technique used to evaluate model accuracy to prevent overfitting. To evaluate the performance, this study used both *k*-fold cross-validation method and hold-out method. The hold-out method consists in simply separating the dataset into a training (60%) and a testing (40%) set. Although the holdout method has been widely used for evaluating model performance, by its nature, it may lead to overfitting since only one dataset is used for training. To overcome this issue, the 10-fold cross-validation is also employed, which ensures reliable model performance while minimizing overfitting problems common in machine learning.

The 10-fold cross-validation process functions as follows. The data set is divided into 10 subsets, and the ANN is trained 10 times. Each time, one of the 10 subsets is used as the test set, and the 9 other subsets are put together to form a training set. The average accuracy across all 10 trials is computed. In this study, the 10-fold cross-validation is applied to the four ANN, and the MNL used to measure the overall model accuracy.

Logit Model

The same training dataset was used to calibrate the MNL, adding two extra dummy variables. The MNL is calibrated with SPSS and Biogeme (*35*). The results suggest that household size, age, walk availability, vehicle fleets, travel time, and travel costs (i.e., gas price and transit fare) are statistically significant (see Table 3).

ANN Model

The input dataset for the ANN consists of 14 input values (i.e., explanatory variables) and four target values (i.e., mode alternatives). The process of data transmission in the neural networks differs by ANN type. For the BPNN model, we adopt the sigmoid activation function. We also use one single hidden layer (using multiple hidden layers did not improve the model performance). The number of hidden neurons was determined through the training process, and we used 21 neurons in the hidden layer.

Table 4. Confusion Matrix for Model Accuracy

	Predicted choice					
Test dataset (BPNN)	Walk (114)	Bike (44)	Auto (958)	CTA (473)	Mode accuracy	Overall accuracy
Observed choice						
Walk (187)	68	13	73	33	36.4%	79.4%
Bike (64)	22	17	11	14	26.6%	
Auto (824)	7	3	781	33	94.8%	
Transit (514)	17	11	93	393	76.5%	
Test dataset (RBFN)	Predicted choice					
	Walk (93)	Bike (27)	Auto (859)	CTA (610)	Mode accuracy	Overall accuracy
Observed choice						
Walk (187)	64	13	67	43	34.2%	78.4%
Bike (64)	2	10	23	29	15.6%	
Auto (824)	17	3	719	85	87.3%	
Transit (514)	10	1	50	453	88.1%	
Test dataset (PNN)	Predicted choice					
	Walk (115)	Bike (45)	Auto (804)	CTA (625)	Mode accuracy	Overall accuracy
Observed choice						
Walk (187)	94	7	36	50	50.3%	82.9%
Bike (64)	6	27	23	8	42.2%	
Auto (824)	2	5	723	94	87.7%	
Transit (514)	13	6	22	473	92.0%	
Test dataset (CPNN)	Predicted choice					
	Walk (140)	Bike (48)	Auto (819)	CTA (582)	Mode accuracy	Overall accuracy
Observed choice						
Walk (187)	97	11	45	34	51.9%	83.3%
Bike (64)	9	31	10	14	48.4%	
Auto (824)	16	4	733	71	89.0%	
Transit (514)	18	2	31	463	90.1%	
Test dataset (MNL)	Predicted choice					
	Walk (108)	Bike (114)	Auto (990)	CTA (377)	Mode accuracy	Overall accuracy
Observed choice						
Walk (187)	55	41	67	24	29.4%	70.5%
Bike (64)	2	21	33	8	32.8%	
Auto (824)	28	13	729	54	88.5%	
Transit (514)	11	27	160	316	61.5%	

To run the kernel-based ANNs (i.e., RBFNs and PNNs), the number of hidden neurons and activation functions have to be determined. For the RBFN, the hidden neurons are activated by the Gaussian function, and the sigmoid function is applied to the output neurons. The number of hidden neurons is usually determined based on the size of data or in a random manner during the training process. In this study, the number of hidden neurons is optimally selected by the algorithm. We set the width of Gaussian function β to 0.2 based on the knowledge of the data and on previous values selected in other studies. For the PNN models, most modeling parameters are similar to RBFNs, except for the activation function of the output neurons, which is not required in PNNs because it is determined by the Bayesian decision rule. For the CPNN, 10 clusters for each alternative are partitioned with *K*-means clustering.

The ANNs are trained using the following four Python open source packages: Scikit-learn (*36*) for BPNN and 10-fold cross-validation, a combination of

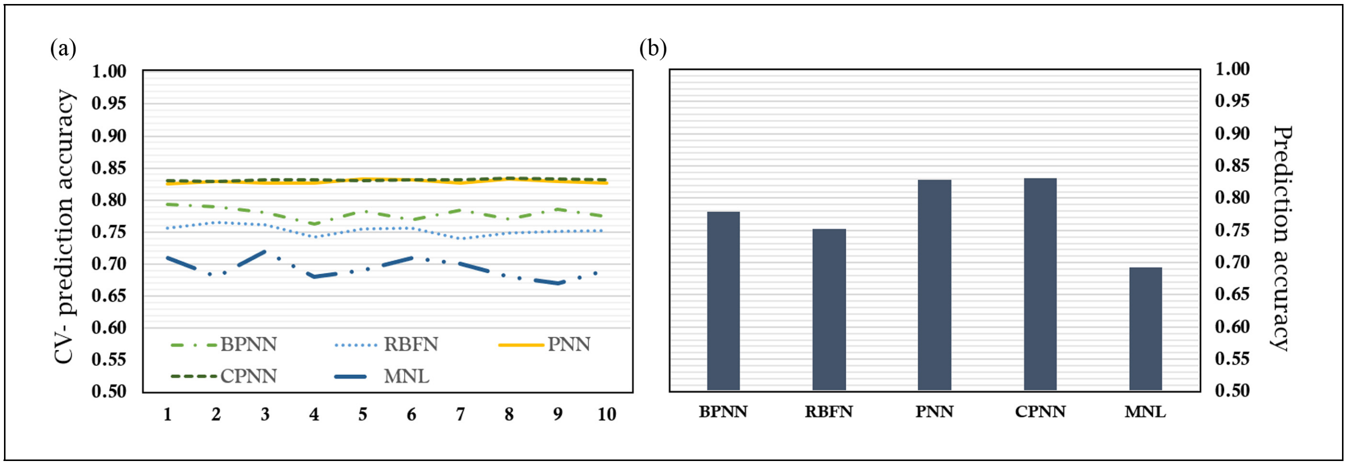

Figure 2. (*a*) Ten-fold cross-validation result for the models; (*b*) comparison of accuracy between the models.

Theano (*37*) and NeuPy (*38*) for PNN and RBFN, and a combination of Scikit-learn (*36*) and NeuPy (*38*) for CPNN. In addition, all ANN models were run on a laptop computer with an average hardware specification (i.e., 6th generation inter processor with 8 GB of RAM). Specifically, computational runtimes for ANN models were similar except for PNN that took 15% longer because of the KDE (specifically for the pattern layer). However, this extra time can be reduced with better hardware specification. In addition, it can be cut down by selecting a limited number of neurons in the pattern layer.

Results

The results of the 10-fold cross-validation for the ANN model are presented in Figure 2*a*. The results indicate that the CPNN model is relatively less sensible over each validation iteration than the BPNN and RBFN. This is because the BPNN and RBFN models are trained with GDM, which may find slightly different minimum values during cross-validation. Furthermore, we compare the average model accuracy between all five models in Figure 2*b*. The results show that the four ANN models achieve better prediction accuracies than the MNL model.

Table 4 presents the confusion matrix, in which each row and each column indicates the observed and predicted the number of travelers for each mode respectively. The overall model accuracy of the four ANNs is around 80%, thus higher than the accuracy of the MNL with 70.5%. While the four ANNs present similar accuracies, the prediction accuracies of each individual mode differ by ANN type. The PNN and CPNN notably show better prediction performance for poorly represented modes. Specifically, the CPNN has slightly better matching rates for the walk and bike modes, in part thanks to the preprocessing with *K*-means clustering.

In addition to the overall accuracy, this study identifies the sensitivity of mode choice decisions from the two most important explanatory attributes: transit costs and auto costs. Unlike traditional regression models, ANN models cannot estimate the impact of explanatory variables on an outcome variable. Thus, sensitivity analysis can be exploited as a sensible option for examining the impact of explanatory variables. Figure 3, *a* and *b* present the result of sensitivity analysis and compare the results by the models for two different variables: auto cost and transit cost. In particular, we can see that BPNN and MNL models are relatively more sensitive to the variations in two variables. As expected, transit users are more likely to change their mode to auto when the price of gas decreases. In contrast, auto users are relatively insensitive to increases in auto costs of 15% or lower.

Conclusion

This article has investigated the feasibility, capability, and performance of four ANN methods in dealing with travel mode choice modeling. The prediction performance of ANN and MNL models has been evaluated by conducting cross-validation tests with the 10-fold cross-validation method. In addition, a confusion matrix has been used to evaluate the matching accuracy between the models. The cross-validation results revealed that the four ANNs achieve better prediction accuracies (around 80%) than the MNL (around 70%). In particular, the CPNN showed the highest performance among the ANN models in part thanks to the *K*-means clustering procedure used. The confusion matrix also showed that the matching rates for bike and walk modes were relatively poor because they were poorly represented in the dataset, although the PNN and CPNN performed best. We suggest that this phenomenon is linked to the fact that ANNs are better able to identify nonlinear relationships in a multivariate survey dataset. Nonetheless,

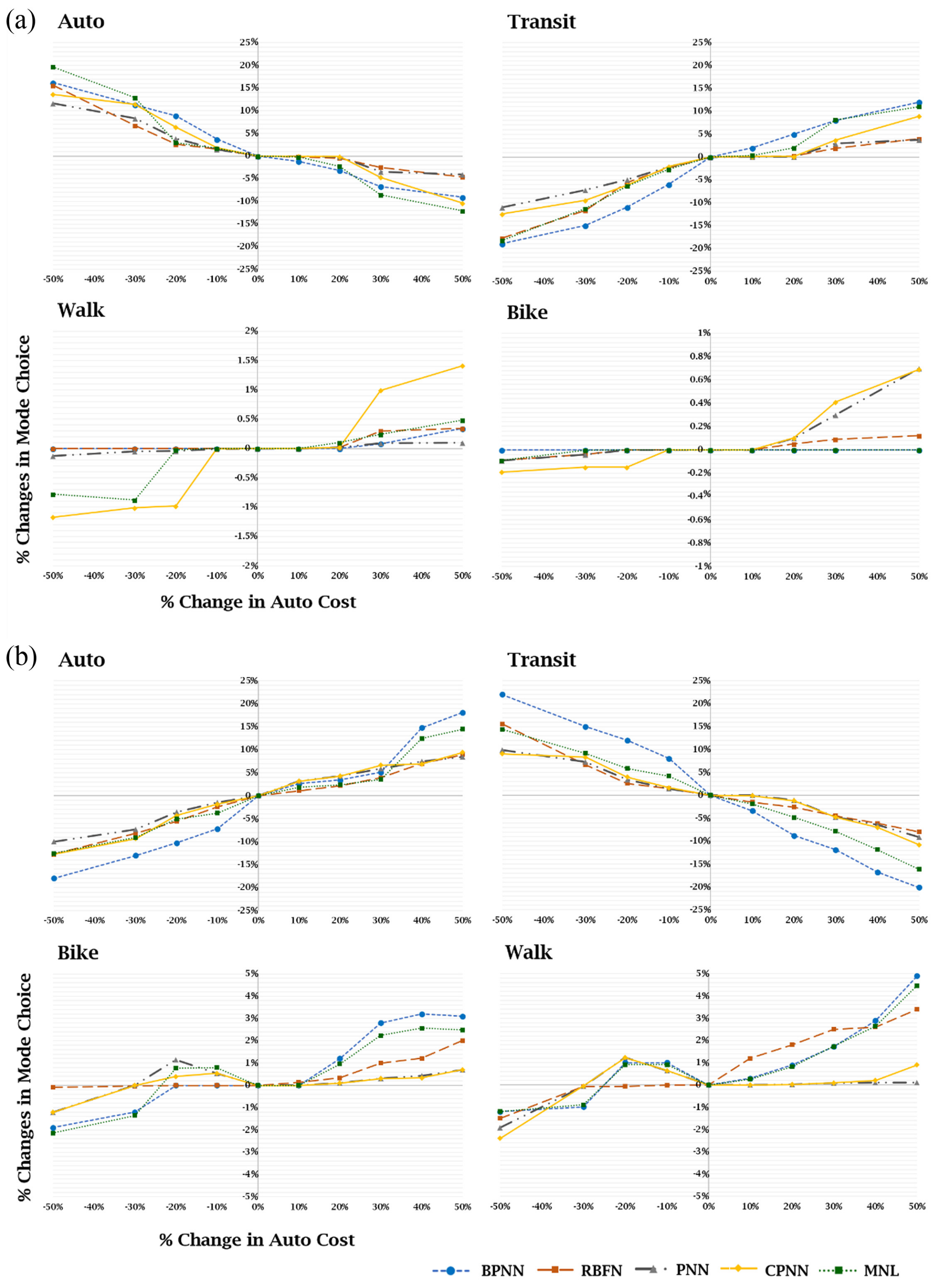

Figure 3. The result of sensitivity analysis—comparison between models based on two scenarios: (*a*) percent changes in auto cost (*b*) percent changes in transit cost.

although the model accuracy of the ANN outperformed the MNL in mode choice modeling, we must acknowledge that ANNs suffer from a lack of interpretability. As a partial strategy to tackle this issue, we conducted a sensitivity analysis for two significant attributes (i.e., transit costs and auto costs) when choosing a mode.

Both the MNL and ANN have advantages and disadvantages. As a parametric approach, logit models are generally used for extracting parameters based on given behavioral patterns. However, when the behaviors being modeled are complex, it becomes difficult to design an accurate logit model. For instance, the MNL is susceptible to unobserved biases, and it is often impractical for identifying the relationship and configuration of explanatory variables. In contrast, ANNs are able to capture nonlinearity and biases in the data by using additional information processing units (e.g., hidden layers). Although ANN is often assimilated to a "black box," its relatively easy applicability and its ability to capture complex patterns make it particularly powerful and promising for the future of mode choice modeling.

For future work, novel non-parametric models have emerged recently (*16*). For instance, deep learning (DL) generally requires more complex algorithmic features, and they are computationally more expensive than ANNs, but in theory, they should be even better able to capture complex and nonlinear relationships in a dataset. Advanced data mining methods such as DL may, therefore, possess a bright future in mode choice modeling in particular, and in travel demand forecasting in general.

Acknowledgments

This research is partly supported by the National Science Foundation (NSF) CAREER award #155173 and by the NSF Cyber-Physical Systems (CPS) award 1646395.

Author Contributions

The authors confirm contribution to the paper as follows: study conception and design: DL, SD; data collection: DL; analysis and interpretation of results: DL, SD, FP; draft manuscript preparation: DL, SD, FP. All authors reviewed the results and approved the final version of the manuscript.

References

1. De Carvalho, M., M. Dougherty, A. Fowkes, and M. Wardman. Forecasting Travel Demand: A Comparison of Logit and Artificial Neural Network Methods. *Journal of the Operational Research Society*, 1998, pp. 717–722.
2. Bentz, Y., and D. Merunka. Neural Networks and the Multinomial Logit for Brand Choice Modelling: A Hybrid Approach. *Journal of Forecasting*, Vol. 19, No. 3, 2000, pp. 177–200.
3. Train, K. E. *Discrete Choice Methods with Simulation*. Cambridge University Press, Cambridge, UK, 2009.
4. Bhat, C. R. Quasi-random Maximum Simulated Likelihood Estimation of the Mixed Multinomial Logit Model. *Transportation Research Part B: Methodological*, Vol. 35, No. 7, 2001, pp. 677–693.
5. Shen, J. Latent Class Model or Mixed Logit Model? A Comparison by Transport Mode Choice Data. *Applied Economics*, Vol. 41, No. 22, 2009, pp. 2915–2924.
6. Kuhn, M., and K. Johnson. *Applied Predictive Modeling*. Springer New York, New York, NY, 2013.
7. James, G., D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning*. Springer, New York, NY, 2013.
8. Derrible, S. Urban Infrastructure is not a Tree: Integrating and Decentralizing Urban Infrastructure Systems. *Environment and Planning B: Planning and Design*, 2016.
9. Ahmad, N., S. Derrible, T. Eason, and H. Cabezas. Using Fisher Information to Track Stability in Multivariate Systems. *Royal Society Open Science*, Vol. 3, No. 11, 2016, p. 160582.
10. Ahmad, N., S. Derrible, and H. Cabezas. Using Fisher Information to Assess Stability in the Performance of Public Transportation Systems. *Royal Society Open Science*, Vol. 4, No. 4, 2017, p. 160920.
11. Ahmad, N., and S. Derrible. Evolution of Public Supply Water Withdrawal in the USA: A Network Approach. *Journal of Industrial Ecology*, Vol. 19, No. 2, 2015, pp. 321–330.
12. Derrible, S., and N. Ahmad. Network-Based and Binless Frequency Analyses. *PloS One*, Vol. 10, No. 11, 2015, p. e0142108.
13. Derrible, S. Complexity in Future Cities: The Rise of Networked Infrastructure. *International Journal of Urban Sciences*, Vol. 21, 2017, pp. 68–86.
14. Cottrill, C. D., and S. Derrible. Leveraging Big Data for the Development of Transport Sustainability Indicators. *Journal of Urban Technology*, Vol. 22, No. 1, 2015, pp. 45–64.
15. Karduni, A., A. Kermanshah, and S. Derrible. A Protocol to Convert Spatial Polyline Data to Network Formats and Applications to World Urban Road Networks. *Scientific Data*, Vol. 3, 2016, p. 160046.
16. Wong, M., B. Farooq, and G.-A. Bilodeau. Latent Behaviour Modelling Using Discriminative Restricted Boltzmann Machines. *Proc., 5th International Choice Modeling Conference*, 2017, pp. 1–18.
17. Bengio, Y., and S. Bengio. Modeling High-Dimensional Discrete Data with Multi-Layer Neural Networks. *Proc., Advances in Neural Information Processing Systems*, 2000.
18. Dougherty, M. A Review of Neural Networks Applied to Transport. *Transportation Research Part C: Emerging Technologies*, Vol. 3, No. 4, 1995, pp. 247–260.

19. Sayed, T., and A. Razavi. Comparison of Neural and Conventional Approaches to Mode Choice Analysis. Vol. 14, No. 1, 2000, p. 23.
20. Xie, C., J. Lu, and E. Parkany. Work Travel Mode Choice Modeling with Data Mining: Decision Trees and Neural Networks. *Transportation Research Record: Journal of the Transportation Research Board*, 2003. 1854: 50–61.
21. Hensher, D. A., and T. T. Ton. A Comparison of the Predictive Potential of Artificial Neural Networks and Nested Logit Models for Commuter Mode Choice. *Transportation Research Part E: Logistics and Transportation Review*, Vol. 36, No. 3, 2000, pp. 155–172.
22. Mohammadian, A., and E. Miller. Nested Logit Models and Artificial Neural Networks for Predicting Household Automobile Choices: Comparison of Performance. *Transportation Research Record: Journal of the Transportation Research Board*, 2002. 1807: 92–100.
23. Golshani, N., R. Shabanpour, S. M. Mahmoudifard, S. Derrible, and A. Mohammadian. Comparison of Artificial Neural Networks and Statistical Copula-Based Joint Models. Travel Behaviour and Society, Vol. 10, 2017, pp. 21–32. doi:10.1016/j.tbs.2017.09.003.
24. Domencich, T. A., and D. McFadden. *Urban Travel Demand: A Behavioral Analysis*. Elsevier, Amsterdam, the Netherlands, 1975.
25. Bishop, C. M. *Pattern Recognition and Machine Learning*. Springer, New York, NY, 2006.
26. Masters, T. *Advanced Algorithms for Neural Networks: A C + + Sourcebook*. John Wiley & Sons, Inc., Hoboken, NJ, 1995.
27. Specht, D. F. Probabilistic Neural Networks. *Neural Networks*, Vol. 3, No. 1, 1990, pp. 109–118.
28. Specht, D. F. Probabilistic Neural Networks for Classification, Mapping, or Associative Memory. *IEEE International Conference on Neural Networks*, San Diego, CA, 1988, pp. 525–532.
29. Parzen, E. On Estimation of a Probability Density Function and Mode. *The Annals of Mathematical Statistics*, Vol. 33, No. 3, 1962, pp. 1065–1076.
30. Cacoullos, T. Estimation of a Multivariate Density. *Annals of the Institute of Statistical Mathematics*, Vol. 18, No. 1, 1966, pp. 179–189.
31. Kim, D. K., D. H. Kim, S. K. Chang, and S. K. Chang. Modified Probabilistic Neural Network Considering Heterogeneous Probabilistic Density Functions in the Design of Breakwater. *KSCE Journal of Civil Engineering*, Vol. 11, No. 2, 2007, pp. 65–71.
32. Monfort, M., B. M. Lake, B. Ziebart, P. Lucey, and J. Tenenbaum. Softstar: Heuristic-Guided Probabilistic Inference. *Proc., Advances in Neural Information Processing Systems*, 2015.
33. Yi, J.-H., J. Wang, and G.-G. Wang. Improved Probabilistic Neural Networks with Self-Adaptive Strategies for Transformer Fault Diagnosis Problem. *Advances in Mechanical Engineering*, Vol. 8, No. 1, 2016, p. 168781401562483.
34. Park, H.-S., and C.-H. Jun. A Simple and Fast Algorithm for K-Medoids Clustering. *Expert Systems with Applications*, Vol. 36, No. 2, 2009, pp. 3336–3341.
35. Bierlaire, M. BIOGEME: A Free Package for the Estimation of Discrete Choice Models. *Proc., Swiss Transport Research Conference*, Monte Verità, Ascona, Switzerland, 2003.
36. Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, and V. Dubourg. Scikit-Learn: Machine Learning in Python. *Journal of Machine Learning Research*, Vol. 12, 2011, pp. 2825–2830.
37. Al-Rfou, R., G. Alain, A. Almahairi, C. Angermueller, D. Bahdanau, N. Ballas, F. Bastien, J. Bayer, A. Belikov, and A. Belopolsky. Theano: A Python Framework for Fast Computation of Mathematical Expressions. *arXiv preprint*, 2016.
38. Shevchuk, Y. *NeuPy: Neural Networks in Python*. http://neupy.com/pages/home.html.

The Standing Committee on Transportation Demand Forecasting (ADB40) peer-reviewed this paper (18-04636).