Improving Market Clearing Price Prediction by Using a Committee Machine of Neural Networks

Jau-Jia Guo, *Student Member, IEEE,* and Peter B. Luh, *Fellow, IEEE*

***Abstract*—Predicting market clearing prices is an important but difficult task, and neural networks have been widely used. A single neural network, however, may misrepresent part of the input-output data mapping that could have been correctly represented by different networks. The use of a "committee machine" composed of multiple networks can in principle alleviate such a difficulty. A major challenge for using a committee machine is to properly combine predictions from multiple networks, since the performance of individual networks is input dependent due to mapping misrepresentation. This paper presents a new method in which weighting coefficients for combining network predictions are the probabilities that individual networks capture the true input-output relationship at that prediction instant. Testing of the New England market cleaning prices demonstrates that the new method performs better than individual networks, and better than committee machines using current ensemble-averaging methods.**

***Index Terms*—Committee machines, energy price forecasting, multiple model approach, neural networks.**

I. Introduction

NEURAL NETWORKS have been widely used in many forecasting problems, including load and market clearing price (MCP) predictions for power systems [1]–[3]. The main reason for their success is that they are capable of inferring hidden relationship (mapping) in data. Such a regression capability comes from the proved property that radial basis function (RBF) and multi-layer perceptron (MLP) networks in theory are universal approximators, and can approximate any continuous function to any degree of accuracy given a sufficient number of hidden neurons [4], [5]. However, in view of reasons such as insufficient input-output data points or too many tunable parameters, in reality a single network often misrepresents part of the nonlinear input-output relationship which could have been more appropriately represented by different neural networks. For example, RBF networks are effective in exploiting local data characteristics, while MLP networks are good at capturing global data trends [6]. Therefore, a committee machine composed of multiple neural networks can in principle alleviate the misrepresentation of input-output data relationship suffered by a single network.

There are two major approaches to obtain predictions for a committee machine. The first approach selects one prediction out of multiple network predictions. For example, the input space of *mixtures of experts* is divided into several regions (subspaces) to which different neural networks are assigned. A gating network decides which network prediction will be selected based on input data [6]. In contrast, the second approach combines predictions of multiple networks. A well-known method is the ensemble-averaging method as depicted in Fig. 1 where predictions of neural networks are linearly combined based on a straight average or the statistics of historical prediction errors [7]–[9].

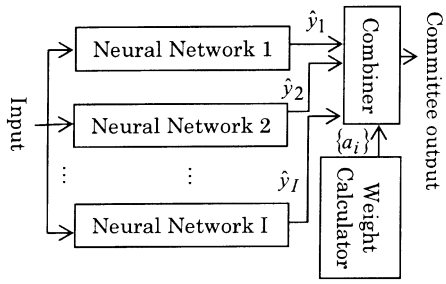

Fig. 1. Schematic of an ensemble-averaging committee machine with network predictions $\hat{y}_i$ and weighting coefficients a_i.

The neural networks in Fig. 1 may be of different kinds, or of the same kind but with different configurations (e.g., different numbers of neurons), or identical but trained with different initial conditions. These networks are trained, perform predictions, and then are updated in a way as if they were stand-alone. A "weight calculator" generates weighting coefficients by which individual predictions are linearly combined in a "combiner." In view that a neural network may misrepresent certain portions of the nonlinear input-output relationship, its prediction accuracy may not be constant. This is evident from MCP prediction results by using RBF and MLP networks in [1] that one network does not always outperform the other. Consequently, combining network predictions is not straightforward. The weighting coefficients of the ensemble-averaging method can reflect the overall historical prediction performance, but do not exploit the information contained in the current input data. The use of the current input data, however, is important because it could be utilized to infer which networks would provide good predictions. It is therefore clear that the ensemble-averaging method does not make the best use of all the available information, and small weights could be assigned to good predictions and large ones to poor predictions, resulting in the poor prediction performance for a committee machine. The purpose of this paper is to present a new method that exploits the current input data and the historical data to calculate weighting coefficients in a weight calculator for a better prediction combination.

Our key idea to exploit the current input data is to estimate the quality of a prediction, that is, the prediction variance that

Manuscript received November 16, 2003. This work was supported in part by the National Science Foundation under Grant ECS-9726577, and by Select Energy Inc. of the Northeast Utilities System. Paper no. PWRS-00642-2003.

The authors are with the Department of Electrical and Computer Engineering, University of Connecticut, Storrs, CT 06279 USA (e-mail: luh@engr.uconn.edu).

Digital Object Identifier 10.1109/TPWRS.2004.837759

is conditioned on the current input and the historical data. Although for a standard Kalman filter with linear dynamics and Gaussian noises, predication variances are independent of input and can be pre-computed offline, neural networks are nonlinear, and prediction variances depend on the current input [9], [10]. Incorporating prediction variances can obtain better weighting coefficients than not using it.

For the completeness of presentation, prediction covariance matrices that will be used for the rest of the paper are briefly derived in Section II for the cases with multiple outputs based on [10]. A forecasting problem using a committee machine as the one shown in Fig. 1 is formulated in Section III where the inadequacy of individual networks due to the misrepresentation of input-output data relationship is first illustrated. Under the Multiple Model (MM) framework [11], it is assumed that the true relationship of any input-output data point follows one of the mappings of multiple networks.

Based on the above formulation, a Bayesian inference-based method to calculate weighting coefficients considering the prediction qualities of networks is presented in Section IV. In our method, a weighting coefficient is shown to be evaluated by the probability that the true input-output relationship at the next prediction instant will follow a specific network mapping. The evaluation involves the combination of mode probabilities and the prediction qualities (that is, prediction covariance matrices) of individual networks where in our method a mode probability is the probability that the true input-output relationship at the current instant follows a specific mapping, and the prediction qualities are used to assess the transition possibility that the true input-output relationship switches from one mapping to another between the current instant and the next prediction instant. In Section V, numerical testing on a simple problem and the MCP prediction using the data from New England power markets demonstrates the advantage of combining multiple networks. Furthermore, the result of the MCP prediction shows that the new method has a better prediction performance than individual networks, and committee machines using current ensemble-averaging methods.

II. Prediction Covariance Matrix

As mentioned in the Introduction, the quality of a prediction is measured by the associated prediction variance. For a multi-output network, the prediction quality is contained in a prediction covariance matrix. The formula for a prediction covariance matrix will be briefly derived in this section. Since the derivation for a single network output is well described in [10], the following derivation for multiple outputs is extended from it. Consider a forecasting problem with a historical input-output data set $\boldsymbol{D}^k = \{\boldsymbol{z}^k, \boldsymbol{t}^k\}$ where $\boldsymbol{z}^k = \{\boldsymbol{z}_1, \ldots, \boldsymbol{z}_k\}$ is a set of noisy inputs and $\boldsymbol{t}^k = \{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_k\}$ is the corresponding set of noisy target outputs. The dimensions of $\boldsymbol{z}_d$ and $\boldsymbol{t}_d$ $(d = 1, \ldots, k)$ are $m_z \times 1$ and $m_t \times 1$, respectively. A noisy input $\boldsymbol{z}_d$ is related to the associated noiseless input $\boldsymbol{x}_d$ by

$$\boldsymbol{z}_d = \boldsymbol{x}_d + \boldsymbol{v}_d \tag{1}$$

where $\boldsymbol{v}_d$ is input noise. To make the analytical estimate possible, $\boldsymbol{v}_d$ is assumed to be independent, identically distributed (i.i.d.), zero-mean, normal, and with a covariance matrix Σ_v. Furthermore, $\boldsymbol{t}_d$ is assumed to be a deterministic function of $\boldsymbol{x}_d$ with additive noise $\boldsymbol{\varepsilon}_d$

$$\boldsymbol{t}_d = \boldsymbol{f}(\boldsymbol{x}_d) + \boldsymbol{\varepsilon}_d. \tag{2}$$

Similarly, $\boldsymbol{\varepsilon}_d$ is assumed to be i.i.d., zero-mean, normal, and with a covariance matrix Σ_ε.

Given a new noisy input $\boldsymbol{z} = \boldsymbol{z}_{k+1}$ and $\boldsymbol{D}^k$, the conditional distribution of a new target output $\boldsymbol{t} = \boldsymbol{t}_{k+1}$ can be written as

$$p(\boldsymbol{t} \,|\, \boldsymbol{z}, \boldsymbol{D}^k) = \int p(\boldsymbol{t} \,|\, \boldsymbol{x}, \boldsymbol{D}^k) p(\boldsymbol{x} \,|\, \boldsymbol{z})\, d\boldsymbol{x} \tag{3}$$

where the new noiseless input $\boldsymbol{x} = \boldsymbol{x}_{k+1}$ is related to $\boldsymbol{z}_{k+1}$ by (1). The distribution $p(\boldsymbol{t} \,|\, \boldsymbol{x}, \boldsymbol{D}^k)$ can be written in terms of the marginal distribution

$$p(\boldsymbol{t} \,|\, \boldsymbol{x}, \boldsymbol{D}^k) = \int p(\boldsymbol{t} \,|\, \boldsymbol{x}, \boldsymbol{w}) p(\boldsymbol{w} \,|\, \boldsymbol{D}^k)\, d\boldsymbol{w}. \tag{4}$$

Given that a neural network can model the true input-output data relationship, $\boldsymbol{f}(\boldsymbol{x}_d)$ in (2) is undertaken by a network output $\boldsymbol{y}(\boldsymbol{x}_d, \boldsymbol{w})$ where $\boldsymbol{w}$ is a set of network weights. Therefore, the first term on the right-hand side of (4) is

$$p(\boldsymbol{t} \,|\, \boldsymbol{x}, \boldsymbol{w}) \propto e^{\frac{-1}{2}\{\boldsymbol{t}-\boldsymbol{y}(\boldsymbol{x},\boldsymbol{w})\}^T \Sigma_\varepsilon^{-1} \{\boldsymbol{t}-\boldsymbol{y}(\boldsymbol{x},\boldsymbol{w})\}}. \tag{5}$$

The term $\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{w})$ in (5) can be linearized around $\mathbf{z}$. Similar to the derivation in [10] with the replacement of scalars and vectors by vectors and matrices, respectively, and with the normal prior assumption for neural weights that encourages a smooth network mapping and is equivalent to a weight-decay regularizer, it can be shown by taking the linearized network output, the Bayes' rule, and the integral over $\boldsymbol{x}$ that

$$p(\boldsymbol{t} \,|\, \boldsymbol{z}, \boldsymbol{D}^k) \propto \int \exp\left[\frac{-1}{2}\{\boldsymbol{t} - \boldsymbol{y}(\boldsymbol{z}, \boldsymbol{w})\}^T \Sigma^{-1} \{\boldsymbol{t} - \boldsymbol{y}(\boldsymbol{z}, \boldsymbol{w})\}\right] \cdot \exp[-S(\boldsymbol{w})]\, d\boldsymbol{w} \tag{6}$$

where

$$S(\boldsymbol{w}) = \frac{1}{2}\sum_{d=1}^{k}[\{\boldsymbol{t}_d - \boldsymbol{y}(\boldsymbol{z}_d, \boldsymbol{w})\}^T \Sigma^{-1} \{\boldsymbol{t}_d - \boldsymbol{y}(\boldsymbol{z}_d, \boldsymbol{w})\}] + \frac{\alpha}{2}\|\boldsymbol{w}\|^2 \tag{7}$$

$$\hat{\boldsymbol{w}}_{\mathrm{MP}} = \arg \mathrm{Min}_{\boldsymbol{w}} S(\boldsymbol{w}) \tag{8}$$

$$\Sigma = \Sigma_\varepsilon + \boldsymbol{H}^T \Sigma_v \boldsymbol{H} \tag{9}$$

$$\boldsymbol{H} = \nabla_{\boldsymbol{x}} \boldsymbol{y}(\boldsymbol{x}, \hat{\boldsymbol{w}}_{\mathrm{MP}})\,|_{\boldsymbol{x}=\boldsymbol{z}}. \tag{10}$$

The first exponential function in (6) results from the new input $\boldsymbol{z}$ and the second one from the historical data $\boldsymbol{D}^k$. The term $S(\boldsymbol{w})$ is the sum-of-squares error function with a regularization term, and the most probable weight vector, $\hat{\boldsymbol{w}}_{\mathrm{MP}}$, is obtained by minimizing $S(\boldsymbol{w})$. Minimizing $S(\boldsymbol{w})$ is the very step of network training.

To make the integral over the weight vector analytically tractable in (6), the first and second order Taylor expansions are applied to $\boldsymbol{y}(\boldsymbol{z}, \boldsymbol{w})$ and $S(\boldsymbol{w})$, respectively. The expansions are performed around $\hat{\boldsymbol{w}}_{\mathrm{MP}}$ and lead to

$$\boldsymbol{y}(\boldsymbol{z}, \boldsymbol{w}) \approx \hat{\boldsymbol{y}}(\boldsymbol{z}, \hat{\boldsymbol{w}}_{\mathrm{MP}}) + \boldsymbol{G}^T(\boldsymbol{w} - \hat{\boldsymbol{w}}_{\mathrm{MP}}) \tag{11}$$

$$S(\boldsymbol{w}) \approx S(\hat{\boldsymbol{w}}_{\mathrm{MP}}) + \frac{1}{2}(\boldsymbol{w} - \hat{\boldsymbol{w}}_{\mathrm{MP}})^T \boldsymbol{A}(\boldsymbol{w} - \hat{\boldsymbol{w}}_{\mathrm{MP}}) \tag{12}$$

where

$$\boldsymbol{G} = \nabla_{\boldsymbol{w}} \boldsymbol{y}(\boldsymbol{z}, \boldsymbol{w})\,|_{\boldsymbol{w}=\hat{\boldsymbol{w}}_{\mathrm{MP}}} \tag{13}$$

$$\boldsymbol{A} = \nabla_{\boldsymbol{w}} \nabla_{\boldsymbol{w}} S(\boldsymbol{w})\,|_{\boldsymbol{w}=\hat{\boldsymbol{w}}_{\mathrm{MP}}}. \tag{14}$$

The first-order term in (12) vanishes because the gradient of $S(\boldsymbol{w})$ at $\hat{\boldsymbol{w}}_{\mathrm{MP}}$ is zero. Substituting (11) and (12) into (6) and evaluating the integral gives the conditional probability distribution function of $\boldsymbol{t}$ as

$$p(\boldsymbol{t}\,|\,\boldsymbol{z}, \boldsymbol{D}^k) = |2\pi\Sigma_t|^{-1/2} \cdot e^{[-\frac{1}{2}(\boldsymbol{t}-\hat{\boldsymbol{y}}(\boldsymbol{z},\hat{\boldsymbol{w}}_{\mathrm{MP}}))^T \Sigma_t^{-1} (\boldsymbol{t}-\hat{\boldsymbol{y}}(\boldsymbol{z},\hat{\boldsymbol{w}}_{\mathrm{MP}}))]} \tag{15}$$

where the mean and covariance matrix of this distribution function are $\hat{\boldsymbol{y}}(\boldsymbol{z}, \hat{\boldsymbol{w}}_{\mathrm{MP}})$ and Σ_t, respectively, and Σ_t is

$$\Sigma_t = \Sigma_\varepsilon + \boldsymbol{G}^T \boldsymbol{A}^{-1} \boldsymbol{G} + \boldsymbol{H}^T \Sigma_v \boldsymbol{H}. \tag{16}$$

The above covariance matrix has three components. The first component comes from output noise, the second one is related to weight uncertainties, and the third one is due to input uncertainties. Furthermore, (15) is the estimate of $p(\boldsymbol{t}|\boldsymbol{z}, \boldsymbol{D}^k)$ by using a neural network, and $\hat{\boldsymbol{y}}(\boldsymbol{z}, \hat{\boldsymbol{w}}_{\mathrm{MP}})$ in (15) is actually the network prediction, and Σ_t is the prediction covariance matrix or the covariance matrix of estimated prediction errors. If a different network is used, $\hat{\boldsymbol{y}}(\boldsymbol{z}, \hat{\boldsymbol{w}}_{\mathrm{MP}})$ and Σ_t in (15) will be changed accordingly.

III. Problem Illustration and Formulation

In this section, the inadequacy of individual networks due to the misrepresentation of part of input-output data relationship is first illustrated, and then the formulation of a forecasting problem using a committee machine is presented. The following example illustrates that RBF and MLP neural networks ([6] and [9]) misrepresent part of the input-output data relationship.

A. Problem Illustration

A nonlinear function is composed of three Gaussian functions, a constant term, and an output noise

$$t = f_1 + 2f_2 + f_3 + 1 + \varepsilon_t$$

where

$$f_1 = e^{-\frac{1}{2}[\frac{(z-4)}{2}]^2}, \quad f_2 = e^{-\frac{1}{2}[\frac{(z-12)}{1.5}]^2}, \quad f_3 = e^{-\frac{1}{2}(z-20)^2}$$
$$z = x + \varepsilon_z, \quad \varepsilon_z \in N(0, 10^{-4}), \text{and } \varepsilon_t \in N(0, 10^{-4}).$$

An RBF network and an MLP network were trained with input z and output t. There were 40 input-output data points with x sampled in [0, 16]. To make network learning for some segments of the relationship more difficult, no data point was generated in [16, 19], and five data points far from the centers of f_1 and f_2 were generated with x uniformly distributed in [19, 21] to contain the relationship of f_3. These 45 data points formed the training set.

According to the best training results obtained, the RBF network with three clusters and the MLP network with seven hidden neurons were used. As summarized in Table I, the RBF network has 0.039 of mean absolute error (MAE) and 2.26% of mean absolute percentage error (MAPE). In contrast, the MAE and MAPE of the MLP network are 0.048% and 2.73%, respectively.

TABLE I
TRAINING RESULTS OF RBF AND MLP NETWORKS

		A: RBF	B: MLP
Training	MAE	0.039	0.048
	MAPE	2.26%	2.73%

* MAE is mean absolute error in unit of $/MWh.
*MAPE is mean absolute percentage error in unit of %.

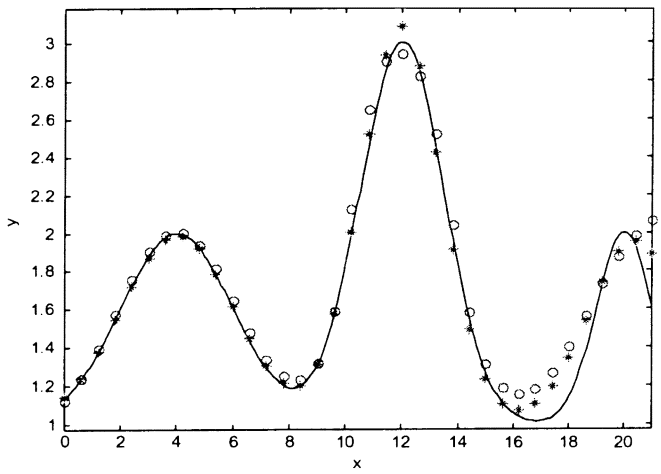

Fig. 2. Plot of function y (the solid curve), the RBF network mapping (stars), and the MLP network mapping (circles).

To further examine network learning, the mappings of these two trained networks are plotted in Fig. 2 with 16 points uniformly sampled in [0, 21]. Fig. 2 clearly shows that for the first region in [0, 16] where there are 40 sampled data points, the RBF mapping overshoots around the segment $x = 12$ where the MLP network learns better. For the second region with x in [16, 19] where no training data points reside, the mappings of both networks are not good because there are no training data points available for the MLP and RBF networks to learn this portion of the relationship. In the third region, with x in [19, 21], where there are five data points, the MLP network does not capture the relationship around $x = 20$, where function f_3 is centered at. The MLP mapping around $x = 20$ has an upward trend and, however, the actual function reverses the rise right after $x = 20$. This is because the MLP network tends to capture the relationship that the majority of data contain, and five out of 45 training data points are not sufficient to learn the relationship of f_3. In contrast, the RBF mapping around $x = 20$ has a downward trend better matching the actual function since this part of the relationship is captured by one of the clusters. Generally speaking, these two network mappings complement each other around $x = 12$ and $x = 20$.

B. Problem Formulation

The inadequacy of a single network can be overcome by using multiple networks. Such a concept has been established in the MM approach [11]. For example, in tracking aircraft, a flight is modeled by a finite number of models with parameters representing selected flying modes. The MM approach assumes that the aircraft follows one of the models at any time instant, and the model results are combined based on radar signals to effectively track the aircraft. The analogy to a forecasting problem using multiple networks is as follows. The true relationship of any input-output data point is assumed to follow one of the mappings of multiple networks, and network predictions are linearly combined to form the outputs of a committee machine. With such an analogy, a forecasting problem using a committee machine that consists of neural networks can be formulated in the following way. Consider a committee machine consisting of I neural networks that are trained, updated, and predict in a way as if they were stand-alone. Given a new input $\boldsymbol{z}$ and the historical input-output data set $\boldsymbol{D}^k$, each of these networks generates an estimated distribution for a new target $\boldsymbol{t}$ at time $k+1$. The estimated distribution is described by (15) where the mean and covariance matrix of the distribution are the network prediction and prediction covariance matrix, respectively. The output $\hat{\boldsymbol{y}}(k+1)$ of a committee machine at time $k+1$ is obtained by a weighted combination of individual network predictions $\hat{\boldsymbol{y}}_i(k+1)$, that is

$$\hat{\boldsymbol{y}}(k+1) = \sum_i a_i(k+1)\hat{\boldsymbol{y}}_i(k+1) \tag{17}$$

where $a_i(k+1)$ are weighting coefficients. The objective is to determine weighting coefficients with the consideration of the prediction qualities of individual networks, i.e., prediction covariance matrices.

IV. Solution Methodology

A. The Bayesian Framework

Since the main interest is to predict a new target output $\boldsymbol{t}$ at the next instant $k+1$ and $\boldsymbol{t}$ that is a vector-valued random variable, it is typical to estimate the probability distribution function (PDF) of $\boldsymbol{t}$ given a new input $\boldsymbol{z}$ and $\boldsymbol{D}^k$. In Section II, the conditional PDF of t estimated by a neural network has been derived. Given that the true relationship of any input-output data point follows one of I network mappings, the true conditional PDF of $\boldsymbol{t}$ is further assumed to be approximated by a linear combination of the estimated conditional PDFs of $\boldsymbol{t}$ generated by I networks. The latter assumption is similar to one suboptimal technique in the MM approach, the *generalized pseudo-Bayesian of first order* method that only considers possible modes at the latest instant to avoid the exponentially increasing number of filters required for an optimal estimate [11]. The mathematical derivation is detailed next.

According to the above, the true distribution of $\boldsymbol{t}$ given $\boldsymbol{z}$ and $\boldsymbol{D}^k$, $p[\boldsymbol{t}|\boldsymbol{z},\boldsymbol{D}^k]$, can be suboptimally decomposed in terms of I estimated PDFs by using the total probability theorem as

$$\begin{aligned} p[\boldsymbol{t}\,|\,\boldsymbol{z},\boldsymbol{D}^k] &= \sum_{i=1}^{I}\{p[\boldsymbol{t}\,|\,M_i(k+1),\boldsymbol{z},\boldsymbol{D}^k] \\ &\quad\cdot p[M_i(k+1)\,|\,\boldsymbol{z},\boldsymbol{D}^k]\} \\ &= \sum_{i=1}^{I}\{c_i(k+1)\cdot p[\boldsymbol{t}\,|\,M_i(k+1),\boldsymbol{z},\boldsymbol{D}^k]\} \end{aligned} \tag{18}$$

where $M_i(k+1)$ denotes the event that at time $k+1$ the mapping of network i is the true input-output relationship, and $c_i(k+1) = p[M_i(k+1)|\boldsymbol{z},\boldsymbol{D}^k]$ is the probability that the true relationship at time $k+1$ will follow the mapping of network i given $\boldsymbol{z}$ and $\boldsymbol{D}^k$. The condition $\sum_i c_i(k+1) = 1$ automatically holds. The term $p[\boldsymbol{t}|M_i(k+1),\boldsymbol{z},\boldsymbol{D}^k]$ is the estimate of $p[\boldsymbol{t}|\boldsymbol{z},\boldsymbol{D}^k]$ by network i and has been derived as in (15) with $\hat{\boldsymbol{y}}(\boldsymbol{z},\hat{\boldsymbol{w}}_{\mathrm{MP}})$ and Σ_t replaced by the prediction $\hat{\boldsymbol{y}}_i(k+1)$ and the prediction covariance matrix $\Sigma_i(k+1)$, respectively. Based on (15) and (18), the conditional mean $\bar{\boldsymbol{t}}(k+1)$ of $p(\boldsymbol{t}|\boldsymbol{z},\boldsymbol{D}^k)$ that meets the minimum mean square error criterion is a linear combination of I network predictions, i.e.,

$$\bar{\boldsymbol{t}}(k+1) = \sum_i c_i(k+1)\hat{\boldsymbol{y}}_i(k+1). \tag{19}$$

Comparing (17) and (19) shows that the output of a committee machine $\hat{\boldsymbol{y}}(k+1)$ would be the conditional mean $\bar{\boldsymbol{t}}(k+1)$ if $a_i(k+1)$ is set equal to $c_i(k+1)$. To yield an estimate of a random variable that meets the minimum mean-square error criterion, $\hat{\boldsymbol{y}}(k+1)$ needs to be $\bar{\boldsymbol{t}}(k+1)$, which results in

$$a_i(k+1) \equiv c_i(k+1) = p[M_i(k+1)\,|\,\boldsymbol{z},\boldsymbol{D}^k]. \tag{20}$$

B. The Determination of Weighting Coefficients

To evaluate $a_i(k+1)$, the total probability theorem will be used. It will be shown later that in our method $a_i(k+1)$ depends on two types of probabilities: a mode probability that the true input-output relationship follows a specific mapping at the current instant, and a mode transition probability describing the transition possibility that the true input-output relationship switches from one mapping to another between the current instant and the next prediction instant given the prediction quality of individual networks.

According to the total probability theorem, $a_i(k+1)$ is decomposed as

$$\begin{aligned} a_i(k+1) = \sum_{i'=1}^{I}\{&P[M_{i'}(k)\,|\,\boldsymbol{z},\boldsymbol{D}^k] \\ &\cdot P[M_i(k+1)\,|\,M_{i'}(k),\boldsymbol{z},\boldsymbol{D}^k]\}. \end{aligned} \tag{21}$$

Since the event $M_{i'}(k)$ is independent of the new input $\boldsymbol{z}$ at time $k+1$, the first term on the right-hand side of (21) is rewritten as

$$P[M_{i'}(k)\,|\,\boldsymbol{z},\boldsymbol{D}^k] = P[M_{i'}(k)\,|\,\boldsymbol{D}^k] \equiv u_{i'}(k) \tag{22}$$

which is the "mode probability," i.e., the probability that the true relationship currently follows the mapping of neural network i, or network i currently is the correct "mode."

To determine the second term on the right-hand side of (21), approximation is made that $\boldsymbol{D}^k$ is summarized by network weights through training, which is similar to the technique used in the MM approach [11]. Given network weights and a new input $\boldsymbol{z}$, predictions and prediction covariance matrices of I networks are obtained. In other words, $\{\boldsymbol{z}, \boldsymbol{D}^k\}$ is summarized by $\{\hat{\boldsymbol{y}}_{i''}(k+1), \sum_{i''}(k+1)\}_{i''=1}^{I}$ where $\hat{\boldsymbol{y}}_{i''}(k+1)$ and $\Sigma_{i''}(k+1)$ are the prediction and the prediction covariance matrix of network i'', respectively. The formula for $\Sigma_{i''}(k+1)$ has been derived as in (16). As a result

$$P[M_i(k+1) \,|\, M_{i'}(k), \boldsymbol{z}, \boldsymbol{D}^k] \approx P\left[M_i(k+1) \,|\, M_{i'}(k), \{\hat{\boldsymbol{y}}_{i''}(k+1), \sum_{i''}(k+1)\}_{i''=1}^{I}\right] \tag{23}$$

is the mode transition probability assessing the likelihood that the true input-output relationship switches from one mapping to another between two consecutive instants given the prediction qualities of individual networks, i.e., prediction covariance matrices. Substituting (22) and (23) into (21) gives

$$a_i(k+1) = \sum_{i'=1}^{I} \left\{ u_{i'}(k) \cdot P\left[M_i(k+1) \,|\, M_{i'}(k), \{\hat{\boldsymbol{y}}_{i''}(k+1), \sum_{i''}(k+1)\}_{i''=1}^{I}\right]\right\}. \tag{24}$$

Equation (24) involves I pairs of a mode probability and a mode transition probability, or I scenarios that portray the transitions from I possible $M_i(k)$ ($i' = 1, \ldots, \mathrm{I}$) at the current instant to $M_i(k+1)$ at the next prediction instant. For each scenario where one specific mapping at the current moment is assumed to be true with a certain probability, prediction qualities are used to evaluate the transition possibility from such a specific mapping to the mapping of network i. The weight calculator combines the evaluation of these I scenarios to determine a weighting coefficient. Since these scenarios are the postulations considered by the weight calculator for the purpose of calculating a weighting coefficient, they do not change the way neural networks operate. The following subsections detail the evaluation of a mode probability and a mode transition probability. The formula of a mode probability can be derived by using the Bayes' rule while a mode transition probability is determined through solving a quadratic optimization problem.

1) Mode Probability Evaluation: To evaluate a mode probability, $u_{i'}(k)$ in (22) is rewritten by using the Bayes' rule as

$$\begin{aligned} u_{i'}(k) &= P[M_{i'}(k) \,|\, \boldsymbol{t}_k, \boldsymbol{z}_k, \boldsymbol{D}^{k-1}] \\ &= \frac{P\lfloor M_{i'}(k), \boldsymbol{t}_k, \boldsymbol{z}_k, \boldsymbol{D}^{k-1}\rfloor}{P[M_{i'}(k), \boldsymbol{z}_k, \boldsymbol{D}^{k-1}]} \cdot \frac{P\lfloor M_{i'}(k), \boldsymbol{z}_k, \boldsymbol{D}^{k-1}\rfloor}{P[\boldsymbol{z}_k, \boldsymbol{D}^{k-1}]} \\ &\quad \cdot \frac{P\lfloor \boldsymbol{z}_k, \boldsymbol{D}^{k-1}\rfloor}{P[\boldsymbol{t}_k, \boldsymbol{z}_k, \boldsymbol{D}^{k-1}]} \\ &= P(\boldsymbol{t}_k \,|\, M_{i'}(k), \boldsymbol{z}_k, \boldsymbol{D}^{k-1}) P[M_{i'}(k) \,|\, \boldsymbol{z}_k, \boldsymbol{D}^{k-1}] \frac{1}{\lambda} \\ &= \frac{1}{\lambda} \Lambda_{i'}(k) P[M_{i'}(k) \,|\, \boldsymbol{z}_k, \boldsymbol{D}^{k-1}] \end{aligned} \tag{25}$$

where $\Lambda_{i'}(k) \equiv p[\boldsymbol{t}_k | M_{i'}(k), \boldsymbol{z}_k, \boldsymbol{D}^{k-1}]$ is the likelihood function of the mapping of network i', and λ is a normalization constant that equals $\sum_{i'} \Lambda_{i'}(k) P[M_{i'}(k) \,|\, \boldsymbol{z}_k, \boldsymbol{D}^{k-1}]$ to maintain $\sum_i u_i(k) = 1$. According to (20) and (24), $p[M_{i'}(k) | \boldsymbol{z}_k, \boldsymbol{D}^{k-1}]$ can be expressed as

$$\begin{aligned} &P[M_{i'}(k) \,|\, \boldsymbol{z}_k, \boldsymbol{D}^{k-1}] \\ &\quad = \sum_{i'''=1}^{I} \left\{ u_{i'''}(k-1) P\left[M_{i'}(k) \,|\, M_{i'''}(k-1), \left\{\tilde{\boldsymbol{y}}_{i''}(k), \tilde{\sum}_{i''}(k)\right\}_{i''=1}^{I}\right]\right\} \end{aligned} \tag{26}$$

where $\tilde{\boldsymbol{y}}_{i''}(k)$ is the output of network i'', and $\tilde{\sum}_{i''}(k)$ is the associated covariance matrix. The second term in the pair of braces in (26) is again a mode transition probability. Combining (25) and (26) gives $u_{i'}(k)$ as

$$u_{i'}(k) = \frac{1}{\lambda} \Lambda_{i'}(k) \cdot \left\{ \sum_{i'''=1}^{I} u_{i'''}(k-1) P\left[M_{i'}(k) \,|\, M_{i'''}(k-1), \left\{\tilde{\boldsymbol{y}}_{i''}(k), \tilde{\sum}_{i''}(k)\right\}_{i''=1}^{I}\right]\right\}. \tag{27}$$

2) Mode Transition Probability Evaluation: Mode transition probabilities appear twice in the above derivation: one in (24) and the other in (27). For a particular pair of (i, i'), $P[M_i(k+1) \,|\, M_{i'}(k), \{\hat{\boldsymbol{y}}_{i''}(k+1), \sum_{i''}(k+1)\}_{i''=1}^{I}]$ in (24) represents the mode transition probability from $M_{i'}(k)$ at k to $M_i(k+1)$ at $k+1$ given prediction covariance matrices of individual networks. As mentioned earlier, a network learns the input-output relationship from data. Given conditioning on the postulated scenario $M_{i'}(k)$ that $(\boldsymbol{z}_k, \boldsymbol{t}_k)$ is in the mapping of network i', the weights of network i' are updated by $(\boldsymbol{z}_k, \boldsymbol{t}_k)$ while those of other networks are not. There are I postulated scenarios because index i' of $M_{i'}(k)$ goes from 1 to I, meaning that every network is conditioned on once. As a result, two versions of optimal weight vectors for each network at any time instant are required: $\hat{\boldsymbol{w}}_{\mathrm{MP},i'}(k)$ for the scenario that $(\boldsymbol{z}_k, \boldsymbol{t}_k)$ at time k is in the mapping of network i, and $\hat{\boldsymbol{w}}^{\mathrm{new}}_{\mathrm{MP},i'}(k)$ for the scenarios that $(\boldsymbol{z}_k, \boldsymbol{t}_k)$ is in other mappings than the mapping of network i. These two versions of weight vectors lead to two sets of predictions and prediction covariance matrices that are used depending on scenarios. The next paragraph will explain how to utilize the current network updating and prediction process to obtain two versions of optimal weight vectors, predictions and prediction covariance matrices.

Fig. 3 depicts one cycle of the network updating and prediction process added with an additional step to yield $\hat{\boldsymbol{w}}^{\mathrm{new}}_{\mathrm{MP},i'}(k)$ that generates another set of predictions and prediction covariance matrices. For neural network 1 (NN1), $\hat{\boldsymbol{w}}_{\mathrm{MP},1}(k)$ is obtained after $\hat{\boldsymbol{w}}_{\mathrm{MP},1}(k-1)$ is updated by $(\boldsymbol{z}_k, \boldsymbol{t}_k)$. In contrast, $\hat{\boldsymbol{w}}^{\mathrm{new}}_{\mathrm{MP},1}(k)$ directly comes from $\hat{\boldsymbol{w}}_{\mathrm{MP},1}(k-1)$ without updating, reflecting the scenario that the true input-output relationship at time k is not in the mapping of network 1. Therefore, two sets of predictions and covariance matrices are obtained for a network by changing versions of weight vectors: $(\hat{\boldsymbol{y}}_1(k+1), \Sigma_1(k+1))$ by using $\hat{\boldsymbol{w}}_{\mathrm{MP},1}(k)$ and $(\hat{\boldsymbol{y}}^{\mathrm{new}}_1(k+1), \sum^{\mathrm{new}}_1(k+1))$ by using $\hat{\boldsymbol{w}}^{\mathrm{new}}_{\mathrm{MP},1}(k)$. Thus,

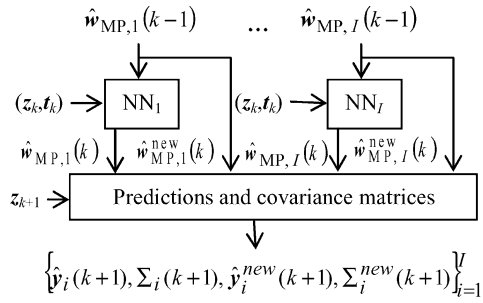

Fig. 3. One cycle of the revised network updating and prediction process.

$P[M_i(k+1)\,|\,M_{i'}(k), \{\hat{\boldsymbol{y}}_{i''}(k+1), \sum_{i''}(k+1)\}_{i''=1}^{I}]$ due to conditioning on $M_{i'}(k)$ can be re-written as

$$P\left[M_i(k+1)\,|\,M_{i'}(k), \{\hat{\boldsymbol{y}}_{i''}(k+1), \sum_{i''}(k+1)\}_{i''=1}^{I}\right] = P\left[M_i(k+1)\;\middle|\;\left\{\hat{\boldsymbol{y}}_{i''}^{\text{new}}(k+1), \sum_{i''}^{\text{new}}(k+1)\right\}_{i''=1, i''\neq i'}^{I}, \hat{\boldsymbol{y}}_{i'}(k+1), \sum_{i'}(k+1)\right]. \quad (28)$$

As learned from Section II, the prediction covariance matrices in (28) also represent the covariance matrices of estimated prediction errors of individual networks. Thus (28) is to determine the probability that the mapping of network i will be the true input-output relationship given the covariance matrices of estimated prediction errors of individual networks. Evaluating (28) is conducted through solving the following optimization problem. Given $\hat{\boldsymbol{y}}_{i'}(k+1), \sum_{i'}(k+1)$, and $\{\hat{\boldsymbol{y}}_{i''}^{\text{new}}(k+1), \sum_{i''}^{\text{new}}(k+1)\}_{i''=1, i''\neq i'}^{I}$, the linear combination of network predictions is

$$\hat{\boldsymbol{y}}(k+1) = \sum_{\substack{i''=1,\\ i''\neq i'}}^{I} p_{i'i''}(k+1)\hat{\boldsymbol{y}}_{i''}^{\text{new}}(k+1) + p_{i'i'}(k+1)\hat{\boldsymbol{y}}_{i'}(k+1). \quad (29)$$

Subtracting both sides of (29) by the actual $\boldsymbol{y}(k+1)$ that is a random variable leads to the combined prediction error as

$$\Delta t(k+1) = \sum_{\substack{i''=1,\\ i''\neq i'}}^{I} p_{i'i''}(k+1)\Delta t_{i''}^{\text{new}}(k+1) + p_{i'i'}(k+1)\Delta t_{i'}(k+1) \quad (30)$$

where $\Delta t_{i'}(k+1)$ and $\Delta t_{i''}^{\text{new}}(k+1)$ $(i''=1,\ldots,\text{I}, i''\neq i')$ are network prediction errors. To minimize the variance of $\Delta t(k+1)$, the statistics of individual network prediction errors at time $k+1$ are required, which is unknown. The only available information is the statistics of estimated network prediction errors, and hence is used instead, implying that the estimated prediction errors are linearly combined.

From the above discussion, we know that for estimated prediction errors $\Delta\hat{t}_{i'}(k+1)$ and $\Delta\hat{t}_{i''}^{\text{new}}(k+1)$ $(i''=1,\ldots,\text{I}, i''\neq i')$, the objective is to find a weighting vector $\boldsymbol{p}_{i'}(k+1) = [p_{i'1}(k+1)\ldots p_{i'I}(k+1)]$ such that the sum

$$\Delta\hat{t}(k+1) = \sum_{\substack{i''=1,\\ i''\neq i'}}^{I} p_{i'i''}(k+1)\Delta\hat{t}_{i''}^{\text{new}}(k+1) + p_{i'i'}(k+1)\Delta\hat{t}_{i'}(k+1) \quad (31)$$

is a vector-valued random variable with the minimum variance. Thus, the optimization problem is

$$\underset{p_{i'}(k+1)}{\text{Min}}\; E\{\Delta\hat{t}(k+1)\Delta\hat{t}^T(k+1)\},$$
$$\text{s.t.}\quad \sum_{i''=1}^{I} p_{i'i''}(k+1) = 1, \quad p_{i'i''}(k+1)\geq 0. \quad (32)$$

Using (31), the objective function can be re-written as

$$E\{\Delta\hat{t}(k+1)\Delta\hat{t}^T(k+1)\} = \boldsymbol{p}_{i'}(k+1)E\left\{\begin{bmatrix}\Delta\hat{t}_1^{\text{new}}(k+1)\\ \vdots \\ \Delta\hat{t}_I^{\text{new}}(k+1)\end{bmatrix} \cdot \begin{bmatrix}\Delta\hat{t}_1^{\text{new}}(k+1)\\ \vdots \\ \Delta\hat{t}_I^{\text{new}}(k+1)\end{bmatrix}^T\right\}\boldsymbol{p}_{i'}(k+1)^T = \boldsymbol{p}_{i'}(k+1)\cdot\boldsymbol{C}\cdot\boldsymbol{p}_{i'}(k+1)^T \quad (33)$$

where $\boldsymbol{C}$ is a covariance matrix for estimated prediction errors of all the networks. The term $\boldsymbol{C}$ is determined by $\boldsymbol{\Sigma}_{i'}(k+1), \{\sum_{i''}^{\text{new}}(k+1)\}_{i''=1, i''\neq i'}^{I}$, and the correlation coefficients $\{r_{ijmn}\}_{i=1,m=1,j<i,n\leq m}^{i=I,m=m_t}$ between estimated prediction errors of the networks that are given as the prior knowledge[1]. In the denotation r_{ijmn}, i and j are network indices, and m and n are network output indices.

The solution $\hat{\boldsymbol{p}}_{i'}(k+1)$ to (32) can be obtained by using the quadratic programming technique. The ith component $\hat{p}_{i'i}(k+1)$ of $\hat{\boldsymbol{p}}_{i'}(k+1)$, which is the weighting coefficient assigned to the estimated prediction error of network i, is the value of (28), that is

$$P\left[M_i(k+1)\;\middle|\;\left\{\hat{\boldsymbol{y}}_{i''}^{\text{new}}(k+1), \sum_{i''}^{\text{new}}(k+1)\right\}_{i''=1, i''\neq i'}^{I}, \hat{\boldsymbol{y}}_{i'}(k+1), \sum_{i'}(k+1)\right] = \hat{p}_{i'i}(k+1). \quad (34)$$

Similarly, the solution to the transition probability in (27) can be denoted as $\hat{q}_{i'''i'}(k)$, that is

$$P\left[M_{i'}(k)\,|\,M_{i'''}(k-1), \{\tilde{\boldsymbol{y}}_{i''}(k), \tilde{\sum}_{i''}(k)\}_{i''=1}^{I}\right] = \hat{q}_{i'''i'}(k). \quad (35)$$

3) Weighting Coefficients in a Committee Machine: Combining (24), (27), (34), and (35), a weighting coefficient $a_i(k+1)$ in a committee machine is

$$a_i(k+1) = \sum_{i'=1}^{I} u_{i'}(k)\hat{p}_{i'i}(k+1) \quad (36)$$

[1]Correlation coefficients can be obtained by using a finite-sample approximation [9].

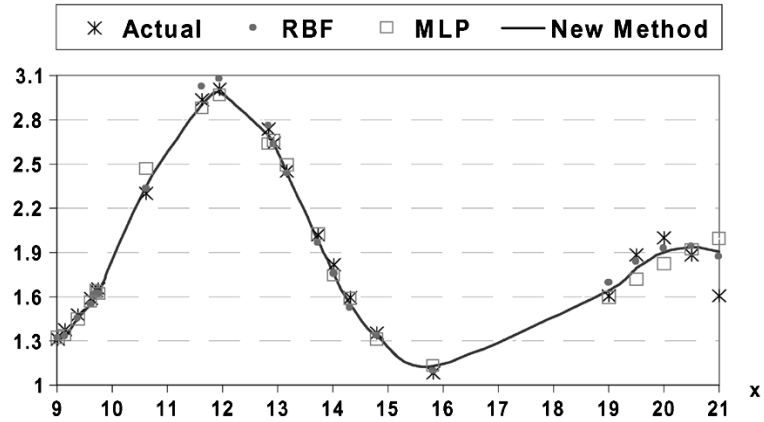

Fig. 4. Plot of actual values and predictions of the RBF network, the MLP network, and the new method.

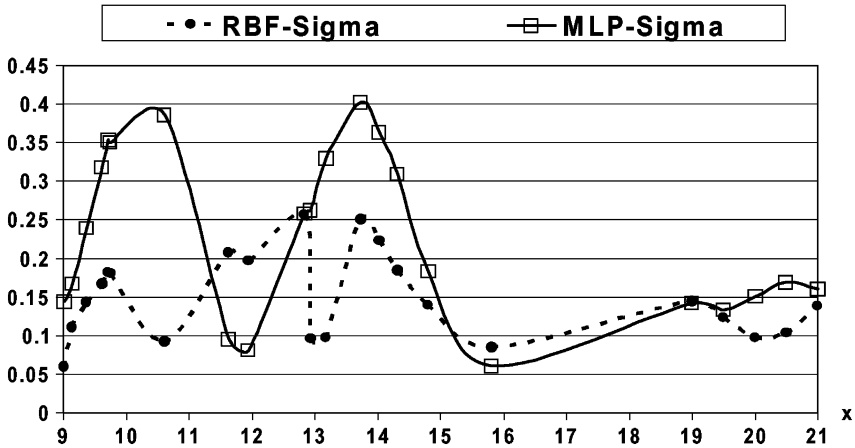

Fig. 5. Prediction standard deviation plot of the RBF and MLP networks.

where

$$u_{i'}(k) = \frac{1}{\lambda}\Lambda_{i'}(k)\left\{\sum_{i'''=1}^{I} u_{i'''}(k-1)\cdot\hat{q}_{i'''i'}(k)\right\} \quad (37)$$

and

$$\lambda = \sum_{i'=1}^{I}\Lambda_{i'}(k)\left\{\sum_{i'''=1}^{I} u_{i'''}(k-1)\cdot\hat{q}_{i'''i'}(k)\right\}. \quad (38)$$

It can be seen from (36) that a weighting coefficient $a_i(k+1)$ is the sum effect of transitions from all the possible network mappings at time k to the mapping of network i at time $k+1$. Furthermore, the prediction qualities of individual networks are incorporated to determine mode transition probabilities. Equation (36), (37), and (38) recursively determine weighting coefficients. After $u_{i'}(0)$ are initialized to be $(1/I)$, where I is the number of neural networks in a committee machine, the recursive sequence starts as follows. First, the mode probabilities at $k, u_{i'}(k)$, are evaluated. Then the weighting coefficients at $k+1, a_i(k+1)$ are determined by combining $u_{i'}(k)$ with the mode transition probabilities, $\hat{p}_{i'i}(k+1)$.

V. Numerical Result and Insights

Two examples are presented in this section to show the advantage of a committee machine over individual networks. The first example is the follow-up to Example 1 to demonstrate the usefulness of the prediction variance information. The second example is a practical application to predict average on-peak-hour MCPs in New England power markets[2] by using three committee machines that consist of RBF and MLP networks. The first one uses the new method, the second one employs a straight average, and the third one utilizes the statistics of historical prediction errors.

A. Study Case 1

This example shows the utilization of the prediction variance information for combining network predictions. The training set in Example 1 serves as the prediction set fed to the trained RBF and MLP networks of Example 1 in this example. The predictions and prediction standard deviations for both networks in the segment with $x = [9, 21]$ are plotted in Figs. 4 and 5, respectively, since the mappings of both networks complement each other around $x = 12$ and $x = 20$.

Fig. 4 shows that the predictions of the RBF and MLP networks are rather close except for the areas with x in [10, 12] and [19, 21] where the accuracy of individual predictions of

[2]New England power markets were restructured eight months ago to follow the concept of Standard Market Design, and have been using locational marginal prices since then. However, MCPs are still used in Independent Electricity Market Operator (IMO) in Ontario.

TABLE II
WEIGHTING COEFFICIENTS ASSIGNED TO RBF AND MCP PREDICTIONS

X	Actual	RBF	MLP	Coeff-RBF	Coeff-MLP
9.01	1.31	1.31	1.32	0.50	0.50
9.14	1.38	1.33	1.33	0.69	0.31
9.38	1.48	1.45	1.45	0.74	0.26
9.62	1.59	1.55	1.57	0.79	0.21
9.72	1.64	1.62	1.64	0.79	0.21
9.74	1.65	1.62	1.62	0.79	0.21
10.61	2.30	2.34	2.46	0.95	0.05
11.64	2.94	3.02	2.88	0.18	0.82
11.95	3.01	3.08	2.97	0.14	0.86
12.84	2.73	2.76	2.63	0.50	0.50
12.94	2.64	2.64	2.66	0.88	0.12
13.17	2.45	2.44	2.49	0.92	0.08
13.74	2.02	1.96	2.02	0.72	0.28
14.02	1.82	1.76	1.74	0.73	0.27
14.32	1.60	1.53	1.59	0.74	0.26
14.81	1.35	1.33	1.31	0.91	0.09
15.81	1.08	1.10	1.13	0.06	0.94
19.00	1.61	1.70	1.59	0.47	0.53
19.50	1.88	1.84	1.72	0.62	0.38
20.00	2.00	1.93	1.83	0.71	0.29
20.50	1.88	1.94	1.92	0.73	0.27
21.00	1.61	1.87	1.99	0.72	0.28

both networks is varying. Along with Fig. 5, it is further shown that the standard deviations of the RBF and MLP network give helpful signals regarding the prediction qualities of both networks in these two areas, which is that better predictions have smaller standard deviations and poorer predictions have larger ones. The new method utilizes the prediction variance information and the resultant weighting coefficients are listed in Table II. The grey area in the table has the weighting coefficients for data points with x in [10, 12] and [19, 21], and shows that proper weightings are assigned to network predictions. Therefore, making use of the prediction variance information benefits the prediction combination.

B. Study Case 2

The prediction for daily average on-peak-hour MCPs for New England power markets was tested in this example. A daily average on-peak-hour MCP is defined by the price averaging MCPs from hour 8 to hour 23. Forecasting average on-peak-hour MCPs is critical because power is often transacted in the form of 16-hour energy blocks. Available data in this testing includes MCPs, loads, surplus, temperatures, oil, and gas prices. The training period is from May 1, 2001 to April 30, 2002 and the prediction period from May 1, 2002 to October 31, 2002. According to the best prediction results obtained, the RBF network uses 23 input factors and six clusters, and the MLP network uses 55 input factors and eight hidden neurons. The list of input factors is shown in Table III, and the target output and all the input factors except Summer and Winter indices are assumed to have the noise of $N(0, 10^{-6})$ after the data is normalized. The performance of the new method is compared not only with that of individual networks, but also with that of the other two other committee machines. One committee machine was implemented with the straight-averaging method, and the other was based on [8], where the correlation matrix to determine weighting coefficients is re-calculated whenever new prediction errors become available. The prediction results are tabulated in Table IV.

TABLE III
LIST OF INPUT FACTORS TO THE RBF AND MLP NETWORKS

	RBF	MLP
Avg on-peak-hour MCP		t-2, t-3, t-4, t-7
Max on-peak-hour MCP		t-2, t-3, t-4, t-7
Min on-peak-hour MCP		t-2, t-3, t-4, t-7
Avg on-peak-hour load	t, t-2, t-7	t, t-2, t-3, t-4, t-7
Max on-peak-hour load	t, t-2, t-7	t, t-2, t-3, t-4, t-7
Min on-peak-hour load	t, t-2, t-7	t, t-2, t-3, t-7
Avg on-peak-hour dry bulb temperature	t, t-1, t-7	t, t-1, t-2, t-7
Max on-peak-hour dry bulb temperature		t, t-1, t-2, t-7
Min on-peak-hour dry bulb temperature		t, t-1, t-2, t-7
Avg on-peak-hour dew point temperature	t, t-1, t-7	t, t-1, t-2, t-7
Max on-peak-hour dew point temperature		t, t-1, t-2, t-7
Min on-peak-hour dew point temperature		t, t-1, t-2, t-7
Henry hub gas price	t, t-7	t
Nymex oil price	t, t-7	t
Projected surplus	t, t-7	t
Summer index	t	t
Winter index	t	t

From the table, it can be seen that the overall MAE and MAPE of the RBF network are 5.46$/MWh and 12.53%, respectively, in contrast with 6.11$/MWh and 13.40% of the MLP network. Actually, the MLP network performs better than the RBF network for the first two months, and the RBF network outperforms the MLP network for the rest of months. For the committee machines using the new method and the straight average, they both outperform individual networks. Comparing overall MAPEs, the new method is better than the RBF and MLP networks by 1.66% and 2.53%, respectively. Comparing overall MAEs, the new method is better than the RBF and MLP networks by 0.60 and 1.25 $/MWh, respectively. Furthermore, the overall performance of the committee machine using the straight-average method is better than that of the committee machine based on the historical prediction performance, and the new method in terms of the overall MAPE is even better than the straight-averaging method and the committee machine based on the historical prediction performance by 0.73% and 1.13%, respectively.

Though the new method is better than individual networks, it as shown from the table did not outperform the MLP and RBF network in June and September, respectively. From the examination of the daily prediction performance in June and September, it shows that improper weighting coefficients for combining network predictions led to the inferior performance of the new method. Improper weighting coefficients occurred in two occasions. The first occasion is that a network with a poorer prediction has a relatively smaller prediction variance than the other network. The second occasion is that when the better-performed model is changed form one network to another, the new method did not adjust weighting coefficients fast enough to keep up with the change. Furthermore, both networks concurrently under-forecasted or over-forecasted prices when the inferior performance of the new method occurred, which makes the new method outperforming individual networks harder.

To explain why the new method has a better prediction performance than individual networks, Fig. 6 shows the August predictions of the RBF network, the MLP network, and the new

TABLE IV
MCP PREDICTION PERFORMANCE COMPARISON

Time	RBF		MLP		Committee Machine (New)		Committee Machine (Straight Averaging)		Committee Machine (Historical Performance)	
	MAE	MAPE	MAE	MAPE	MAE	MAPE	MAE	MAPE	MAE	MAPE
May-02	4.10	11.72	3.88	10.72	3.59	10.09	3.90	10.97	4.27	11.75
Jun-02	5.01	15.91	3.97	13.08	4.31	13.96	4.28	14.04	4.14	13.60
Jul-02	5.44	11.66	6.56	14.11	4.72	9.70	5.59	11.64	6.10	13.01
Aug-02	9.27	16.61	11.66	19.85	7.38	11.86	7.91	13.24	8.22	14.05
Sep-02	5.40	11.98	6.85	14.91	5.67	12.46	5.74	12.73	5.63	12.50
Oct-02	3.55	7.39	3.71	7.74	3.50	7.28	3.42	7.10	3.43	7.12
Overall	5.46	12.53	6.11	13.40	4.86	10.87	5.14	11.60	5.30	11.99

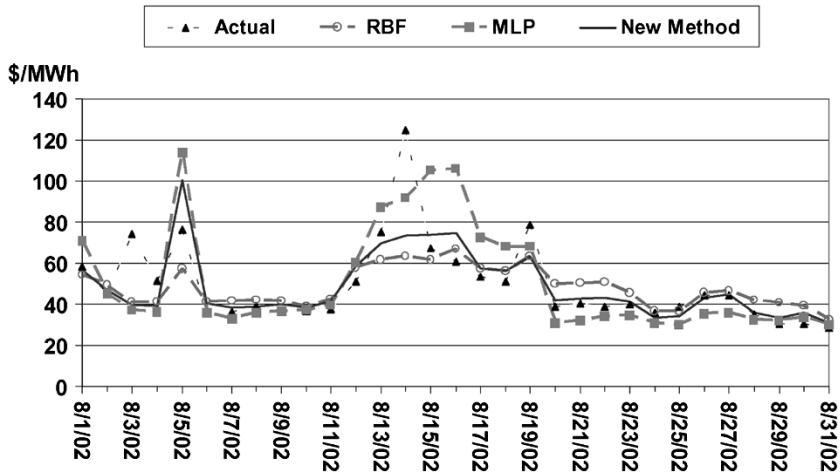

Fig. 6. Prediction plot of the RBF and MLP networks, and the committee machine using the new method in August 2002.

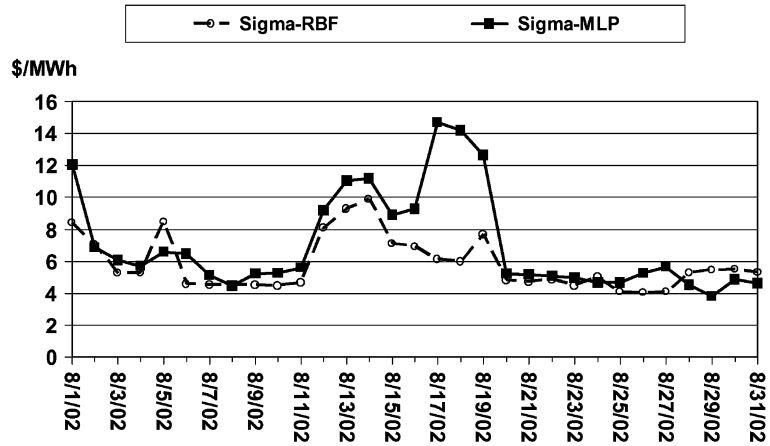

Fig. 7. Standard deviation plot of the RBF and MLP networks in August 2002.

method. Fig. 7 is the plot of the associated standard deviations of RBF and MLP predictions.

It can be seen from Fig. 6 that the actual prices in August changed drastically. During that period of time, the predictions of the RBF and MLP networks were rather distinct. Fig. 7 shows the prediction standard deviations of both networks which are interweaved. Utilizing the information of prediction variances, the new method is able to assign appropriate weighting coefficients to network predictions for most of days. One more thing is worth being noted. With the assumption that input and output noises are i.i.d., zero-mean, and normal, the distribution of a prediction error estimated by a neural network in Section II is approximated to be normal. As shown in Fig. 8, the histograms of the RBF and MLP prediction errors are normal, which means that the assumption for input and output noises is acceptable.

VI. CONCLUSION

This paper applies a committee machine to a forecasting problem, and develops a new method under the *Multiple Model*

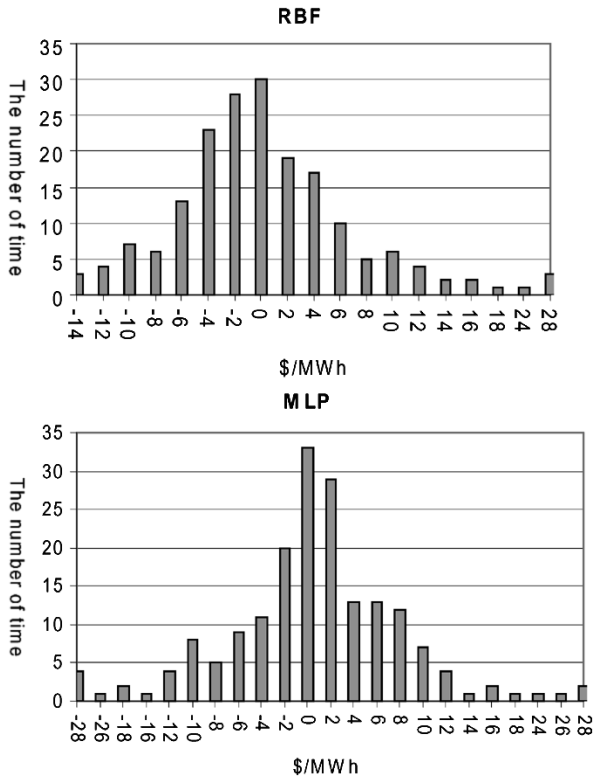

Fig. 8. Histograms of the RBF and MLP prediction errors.

framework to determine weighting coefficients. The weighting coefficients assigned to networks is shown to be the probabilities that individual networks capture the true input-output relationship at that prediction instant. The prediction qualities (that is, prediction covariance matrices) of individual networks are incorporated into the evaluation of the weighting coefficients. The testing results show that the new method not only performs better than the committee machines using current ensemble-averaging methods but also outperforms individual neural networks.

REFERENCES

[1] J. J. Guo and P. B. Luh, "Selecting input factors for clusters of gaussian radial basis function network to improve market clearing price prediction," *IEEE Trans. Power Syst.*, vol. 18, pp. 665–672, May 2003.

[2] A. G. Bakirtzls, V. Petridls, S. J. Klartzis, M. C. Alexladls, and A. H. Malssls, "A neural network short term load forecasting model for the Greek power system," *IEEE Trans. Power Syst.*, vol. 11, pp. 858–863, May 1996.

[3] H. R. Kassaei, A. Keyhani, T. Woung, and M. Rahman, "A hybrid fuzzy, neural network bus load modeling and predication," *IEEE Trans. Power Syst.*, vol. 14, pp. 718–724, May 1999.

[4] E. J. Hartman, J. D. Keeler, and J. M. Kawalski, "Layered neural networks with gaussian hidden units as universal approximators," *Neural Networks*, vol. 35, no. 2, pp. 210–215, 1990.

[5] K. Hornik, M. Stinchcombe, and H. White, "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks," *Neural Networks*, pp. 551–560, 1990.

[6] S. Haykin, *Neural Networks: A Comprehensive Foundation*. Englewood Cliffs, NJ: Prentice-Hall, 1985.

[7] X. Yao and Y. Liu, "Making use of population information in evoltionary artificial neural networks," *IEEE Trans. Syst., Man, Cybern. B*, vol. 28, pp. 417–425, June 1998.

[8] B. T. Zhang and J. G. Joung, "Time series prediction using committee machines of evolutionary neural trees," in *Proc. of the Congress on Evolutionary Computation (CEC'99)*, vol. 1, July 1999, pp. 281–286.

[9] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford, 1985, pp. 385–399.

[10] W. A. Wright, "Bayesian approach to neural-network modeling with input uncertainty," *IEEE Trans. Neural Networks*, vol. 10, pp. 1261–1270, Nov. 1999.

[11] Y. Bar-Shalom and X. Li, *Estimation and Tracking: Principles, Techniques, and Software*. Norwood, MA: Artech House, 1993, pp. 450–453.

Jau-Jia Guo (S'00) received the B.S. degree in engineering science from National Cheng-Kung University, Tainan, Taiwan, R.O.C., in 1993, and the M.S. degrees in physics and in electrical and computer engineering from the University of Connecticut, Storrs, in 1997 and 2000, respectively. He is currently pursuing the Ph.D. degree in the Electrical and Computer Engineering Department, University of Connecticut.

Peter B. Luh (M'80–SM'91–F'95) received the B.S. degree in electrical engineering from National Taiwan University, Taipei, Taiwan, R.O.C., in 1973, the M.S. degree in aeronautics and astronautics engineering from Massachussetts Institute of Technology, Cambridge, in 1977, and the Ph.D. degree in applied mathematics from Harvard University, Cambridge, in 1980.

Since 1980, he has been with the University of Connecticut, Storrs, where he is currently a SNET Endowed Professor of Communications and Information Technologies with the Department of Electrical and Computer Engineering, and the Director of the Booth Research Center for Computer Applications. His major research interests include schedule generation and reconfiguration for manufacturing and power systems.

Dr. Luh is Editor-in-Chief of the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING. He was previously the Editor-in-Chief of theIEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, and an Associate Editor of the IEEE TRANSACTIONS ON AUTOMATIC CONTROL.