Article

Improved Deep Learning Model Based on Self-Paced Learning for Multiscale Short-Term Electricity Load Forecasting

Meiping Li ^{*} , Xiaoming Xie and Du Zhang

Faculty of Information Technology, Macau University of Science and Technology, Macau 999078, China; kevinmingxie@outlook.com (X.X.); duzhang@must.edu.mo (D.Z.)

* Correspondence: meipinglimust@gmail.com

Abstract: Electricity loads are basic and important information for power generation facilities and traders, especially in terms of production plans, daily operations, unit commitments, and economic dispatches. Short-term load forecasting (STLF), which predicts power loads for a few days, plays a vital role in the reliable, safe, and efficient operation of a power system. Currently, two main challenges are faced by existing STLF prediction models. The first involves how to fuse multiscale electricity load data to obtain a high-performance model and remove data noise after integration. The second involves how to improve the local optimal solution despite the sample quality problem. To address the above issues, this paper proposes a multiscale electricity load data fusion- and STLF-based short time series prediction model built on a sparse deep autoencoder and self-paced learning (SPL). A sparse deep autoencoder was used to solve the multiscale data fusion problem with data noise. Furthermore, SPL was utilized to solve the local optimal solution problem. The experimental results showed that our model was better than the existing STLF prediction models by more than 15.89% in terms of the mean squared error (MSE) indicator.

Keywords: short-term load forecasting (STLF); autoencoder; self-paced learning (SPL)

Citation: Li, M.; Xie, X.; Zhang, D. Improved Deep Learning Model Based on Self-Paced Learning for Multiscale Short-Term Electricity Load Forecasting. *Sustainability* **2022**, *14*, 188. <https://doi.org/10.3390/su14010188>

Academic Editor: Alberto Jesus Perea-Moreno

Received: 29 November 2021

Accepted: 22 December 2021

Published: 24 December 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Electricity loads are basic and important information for power generation facilities and traders [1], especially in terms of production plans, daily operations, unit commitments, and economic dispatches [2]. Currently, based on the length of the forecasting time window, three types of load forecasting models are available [3]. First, long-term load forecasting can be used to predict the power load situations of factories and infrastructure for a window of several years and help investors make decisions [4]. Second, medium-term load forecasting can generally be used to predict the power load situation of a target area for time periods ranging from a few days to a few months [5]. Third, short-term load forecasting (STLF), which can generally forecast power loads for only a few days or a few hours, is generally used for real-time power generation control, safety analysis, and energy transaction planning [6]. STLF can be performed at the national, regional, or microgrid level [7]. The supply–demand balance rule applies to the electricity market, with electricity prices increasing during periods with high electricity loads and decreasing during periods with low electricity loads (such as nights, weekends, and holidays) [8]. It is worth noting that the power load is determined on an hourly basis. In a large power plant, a production plan is mostly carried out every day. In summary, STLF plays a vital role in managing the operations of the electricity market. In addition, due to the COVID-19 pandemic, the prices of the raw materials for use in electricity production have risen sharply [9], which has made the supplies of electricity in many countries increasingly tight [10]. This makes STLF more important [11].

In STLF, selecting a suitable training set and building an optimized prediction model are the keys to improving the resulting prediction accuracy. Therefore, researchers have proposed many models in the past few decades. The existing STLF models are mainly

based on artificial intelligence (AI) methods [12]. At present, the main AI models for STLFL include deep learning [13], support vector machines (SVMs) [14], and genetic models [15]. For example, Barman et al. proposed a hybrid FA–SVM model for short-term load forecasting [16]. Liu et al. proposed a KF–BA–SVM model to predict the data of a substation in South China [17]. The latest research has shown that multilayer perceptron (MLP) [18] and long short-term memory (LSTM) models [19] perform STLFL better than existing models [20]. For example, Kong et al. proposed an LSTM model that can be used for household electric load forecasting [21]. Mujeeb et al. proposed a power load forecasting method based on LSTM [22]. An article by Kontogiannis et al. showed that the power load model based on an MLP is better than a convolutional neural network (CNN) model and an LSTM model [23].

However, two main challenges are still faced by existing STLFL models. The first involves how to fuse multiscale load data to achieve a higher performance than that of existing models and solve the data noise problem after integration. Compared with a single-scale training model, according to existing research, a multiscale STLFL model can achieve a higher prediction performance [24]. However, multiscale data increase the noise in the given data. This increases the difficulty of building an STLFL model. The second challenge is how to prevent the STLFL model from falling into local optimal solutions. Sliding window segmentation causes sample quality problems. The use of all samples for training makes it easy for the model to fall into a local optimum.

This paper proposes a multiscale fusion-based STLFL model built on a sparse deep autoencoder and self-paced learning (SPL). The sparse deep autoencoder proposed in this paper is a supervised neural network for the fusion and denoising of multiscale STLFL data. The model includes two parts: an encoder and a decoder [25]. The regularization term is added to existing depth encoders to equip the model with sparse coding capabilities. SPL is a learning mechanism that was proposed in recent years in the field of ML; it is inspired by the learning processes of humans and animals, which operates from easy to difficult. SPL embeds the difficulty of course learning into the utilized optimization model while updating the model parameters based on the current sample ranking and updating the difficulty rankings of the samples based on the induced learning effect [26]. The goal of SPL is to solve the problems of low model accuracy and convergence difficulty caused by sample quality problems [27].

In summary, this article provides the following three main contributions:

- The SPL strategy gradually incorporates samples into the developed model from simple to complex. This paper innovatively proposes combining SPL with the MLP method, which can effectively avoid local optimal solutions and further improve the prediction performance of the model.
- To the best of our knowledge, AE–SPLMLP is the first multiscale power load forecasting model that incorporates a sparse deep autoencoder and the MLP method based on SPL into a computational framework.
- The obtained experimental results show that compared with the support vector regression (SVR), gradient boosting decision tree (GBDT), extreme gradient boosting (XgBoost), light gradient boosted machine (LightGBM), and LSTM models, the AE–SPLMLP model achieves improvements of 6.88–99.66%.

The organization of the paper is as follows. In Section 2, the factors that need to be considered in the study of power load forecasting are introduced, such as weather and temperature, as well as the data used in this paper. Then, a sparse deep autoencoder and the technique of combining SPL with the MLP method are described. Section 3 presents the results of an experiment. The discussion and conclusion are provided in Section 4.

2. Datasets and Methods

2.1. Factors and Datasets

In this paper, the utilized data come from a standard dataset provided by the Ninth “China Electrical Engineering Society Cup” National Undergraduate Electrical Engineering Mathematical Modelling Competition; these data can be downloaded from <http://shumo.com>.

neepu.edu.cn/index.php/Home/Zxdt/news/id/3.html \ https://github.com/Meiping-Li/Electricity_load_data.git (accessed in October 2021).

The influencing factors of electricity load changes mainly include natural factors and social factors [28]. The characteristic variables related to natural factors and social factors mainly include temperature, weather, holidays, and working days [29]. In addition, the current load change is closely related to the historical load. Therefore, the historical load values also form an important feature variable [30]. The characteristic variables used in this article are shown in Equation (1).

$$X = [N_1, S_1, L_1, \dots, N_n, S_n, L_n] \quad (1)$$

where N_n , S_n , and L_n represent the natural, social, and historical load features, respectively, and n represents the number of variables in each category. This paper mainly uses natural factors and historical loads. Natural factors include daily maximum temperatures, daily minimum temperatures, daily average temperatures, daily relative humidity levels, and daily rainfall levels. The data include the power load values of a target area (sampled once every 15 min, 96 times a day). Since this article is based on short-term power load forecasting at daily intervals, the daily peak is used as the power load value for each day in the historical load data [31].

To predict power loads more accurately, this paper divides the natural factors and historical load data into multiple scales. The intervals of 5, 8, and 12 days are used (the step size is 1 day) to divide the dataset into multiple scales. For example, to predict the power load value on the 13th day, the natural factors and historical load data of the previous 5 days, 8 days, and 12 days need to be used as input features for prediction. Finally, the segmented dataset is divided into a training set and a testing set at a ratio of 7:3.

2.2. Deep Autoencoder

The multiscale power load data contain considerable noise, and the greater the scale, the more noise the dataset contains. Moreover, reasonably fusing multiscale data is also a challenge. Therefore, this paper uses a deep autoencoder to denoise the multiscale power load data and extract the intermediate hidden layer to fuse these multiscale data. A deep autoencoder is an unsupervised learning model based on backpropagation and gradient descent methods [25]. It uses the input data X themselves for supervision, learns the underlying mapping relationship, and finally obtains a reconstructed output X^R . Since the STLF data themselves have noise, it is normal for there to be a certain difference between the reconstructed X^R and the original data X . This paper also adds sparse regularization in the process of data reconstruction to remove data noise.

The sparse deep autoencoder in this article contains two main parts: an encoder and a decoder.

The function of the encoder is to encode the high-dimensional input X into a low-dimensional hidden variable h so that the neural network can learn the most informative features. The encoding process of the original data X from the input layer to the hidden layer is as follows:

$$h = \sigma(W_1 x + b_1). \quad (2)$$

The function of the decoder is to restore the hidden variable h of the hidden layer to the initial dimensionality. The best state is that in which the output of the decoder can perfectly or approximately restore the original input; that is, $X^R \approx X$.

$$\hat{x} = \sigma(W_2 x + b_2). \quad (3)$$

The optimization objective function is as follows:

$$\text{Minimize Loss} = \text{dist}(X, X^R) \quad (4)$$

where $dist(.,.)$ is the distance measurement function between the two data versions; this is usually the mean squared error (MSE).

2.3. Sparse Coding

Generally, the purpose of an autoencoder is to learn the inherent laws of the input data and compress the data. At the same time, obtaining the internal structure and laws of the input data is also very important. Therefore, a sparsity restriction is added for the self-encoder; that is, when the model is trained, some hidden layer nodes are activated and others are suppressed so that the entire self-encoder becomes sparse.

If the hidden layer uses a sigmoid activation function, a hidden layer output of 1 means that the node is “active” and a hidden layer output of 0 means that the node is “inactive”. In this case, this paper introduces Kullback–Leibler (KL) divergence to measure the similarity between the average activation output of a hidden layer node and the sparsity ρ , as follows:

$$KL(\rho \parallel \hat{\rho}_j) = \rho \log \frac{\rho}{\hat{\rho}_j} + (1 - \rho) \log \frac{1 - \rho}{1 - \hat{\rho}_j}. \quad (5)$$

where $\hat{\rho}_j = \frac{1}{m} \sum_{i=1}^m [a_j^{(2)} x^{(i)}]$, m is the number of training samples, and $a_j^{(2)}$ is the response output of the j -th node in the hidden layer to i samples. Generally, this paper sets the sparsity coefficient $\rho = 0.05$. The greater the KL divergence, the greater the difference between ρ and $\hat{\rho}_j$, and a KL divergence equal to 0 means that the two are completely equal ($\rho = \hat{\rho}_j$).

Consequently, the KL divergence is added as a regular term into the loss function to constrain the sparse rows of the entire autoencoder network:

$$J_{sparse}(W, b) = J(W, b) + \beta \sum_{j=1}^{s2} KL(\rho \parallel \hat{\rho}_j). \quad (6)$$

2.4. SPL

Consider a standard supervised learning problem. Given a training dataset $D = \{x_i, y_i\}_{i=1}^N$, where $x_i \in X$ represents a sample (X represents the sample space), $y_i \in Y$ represents the corresponding truth (Y represents the label space), and N is the number of samples, the goal of ML is to learn a prediction function $f_w : X \rightarrow Y$, where w is the parameter of the prediction function. After obtaining this decision function, any new sample can be predicted. The solution of this problem generally requires the following optimization model:

$$\min \sum_{i=1}^N \mathcal{L}_{Train}(f_w(x_i), y_i + R_w) \quad (7)$$

where $\mathcal{L}_{Train}$ is the error term of the training dataset. Its function is to measure the degree of difference between the prediction and the true value. A commonly used error function is the MSE. R_w represents the regular term.

The basic execution mode of SPL is as follows. In a traditional machine learning model, the weight variable $v = \{v_1, v_2, \dots, v_N\} \in [0, 1]^N$, which measures the difficulty of each sample, is embedded, and the following optimization problem is solved. Finally, the parameters v and w are jointly optimized to accomplish the learning process from easy to difficult [8].

$$\min_{w \in \Gamma, v \in [0, 1]^N} \sum_{i=1}^N (v_i \mathcal{L}_{Train}(f_w(x_i), y_i) + g(v_i, \lambda)) \quad (8)$$

where λ is the age parameter and $g(v_i, \lambda)$ denotes regular self-stepping, which is used to encode the SPL mode from easy to difficult. For the definition and conditions of SPL that must be met, please refer to references [9,10]. Since R_w has no essential effect on the analysis of the self-stepping regularity, this term is omitted in this research. Subsequently,

to simplify the symbolic representation, the error of the i -th sample $\mathcal{L}(f_w(x_i), y_i)$ is often used, which is abbreviated as ℓ_i .

2.5. Combining SPL with an MLP

A backpropagation MLP generates an optimal network structure through the reverse training of the model based on the input sample data. The training process of an MLP is mainly divided into two stages. The first stage is mainly conducted among the input layer, hidden layer, and output layer. After the input vector is linearly summed, the selected activation function is used to output a result. The second stage is mainly based on a loss function. The model uses the stochastic gradient descent method for directional propagation from the input layer to the hidden layer and then to the output layer. Finally, the weights and biases among the output layer, hidden layer, and input layer are updated. For a given training dataset $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, the loss function of an MLP is as follows:

$$\min_{w,b} \frac{1}{2} \sum_{i=1}^n (y_i - \sigma(wx_i + b))^2. \quad (9)$$

where $x_n \in \mathbb{R}^N$ and $y_n \in \mathbb{R}^N$ are the sample or feature space and the label space, respectively.

To overcome the shortcomings of an MLP easily falling into local optima, this paper proposes the AE-SPLMLP model by combining SPL and the MLP method (see Figure 1). This model selects samples in order from simple to complex by means of hard thresholds. For a given training dataset $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$, the final loss function of the AE-SPLMLP model is as follows:

$$\min_{w,b,v} \frac{1}{2} \sum_{i=1}^n \left(v_i L(f(x_i), y_i) + \frac{1}{k} v_i \right) \quad (10)$$

where $L(f(x_i), y_i) = (y_i - \sigma(wx_i + b))^2$.

Figure 1. (A) A sparse deep autoencoder is used for multiscale data fusion. (B) An SPL framework is used for regression-based prediction.

To solve the problem in Equation (10), an alternative optimization strategy (AOS)-based algorithm can be used to update the parameters w and v , where the input is the training dataset D and the output is the prediction result $\hat{y}$; the solving process of the AE-SPLMLP model is as shown by the flowchart in Figure 2. This detailed updating process is as follows:

Step 1. Fix the parameter w^* and solve for v ; that is, solve the following problem:

$$v_i^* = \arg \min_{v \in [0,1]} v_i L(f_{w^*}(x_i), y_i) + \frac{1}{k} v_i, i = 1, 2, \dots, n. \quad (11)$$

Step 2. Fix the parameter v^* and solve for w ; that is, solve the following problem:

$$w^* = \arg \min_w \sum_{i=1}^N v_i^* L(f_w(x_i), y_i). \quad (12)$$

Step 3. After updating v and w , increase the value of λ to cause more samples to enter the learning process.

Figure 2. Flowchart of the AE-SPLMLP model.

3. Results

In this section of the paper, our electricity load forecasting model is evaluated on data obtained from the Energy Information Administration. Moreover, this paper selects five classic machine learning and deep learning models, including the SVR, GBDT, XgBoost, LightGBM, and LSTM models, for comparison with AE-SPLMLP (our proposed method).

3.1. Implementation Details and Performance Metrics

This study used an autoencoder to denoise the multiscale data and construct three hidden layers. The numbers of neurons in these layers were set to 200, 100, and 200. For the SPLMLP model, this study used a hidden layer, the number of hidden layer neurons was set to five, the activation function was set as a sigmoid function, and the number of iterations was 500. The detailed hyperparameter settings of the different models are shown in Table 1.

Table 1. The hyperparameters considered for different models.

Model	Model Hyperparameters	Considered Values
Autoencoder	Number of hidden layers	3
		200
	Number of neurons	100
		200
SPLMLP	Number of hidden layers	1
	Number of neurons	5
	Activation function	sigmoid
	Number of iterations	500

The MSE, root mean square error (RMSE), coefficient of variation of the RMSE (CV-RMSE), and mean absolute error (MAE) were applied as four evaluation indicators in this subsection. The definitions of these evaluation indicators are shown in Equations (13)–(16).

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad (13)$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2}, \quad (14)$$

$$\text{CV-RMSE} = \frac{\sqrt{\frac{1}{n-1} \sum_{i=1}^n (y_i - \hat{y}_i)^2}}{\frac{\sum_{i=1}^n y_i}{n}}, \quad (15)$$

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|. \quad (16)$$

3.2. The Effects of the Hyperparameters

The hyperparameter settings had a great influence on the performance of the final model. This study focused on three specific hyperparameters: the learning rate, the correction coefficient, and the number of hidden layer neurons. Two indicators, the MSE and MAE, were used to find the optimal hyperparameters. The hyperparameters that could minimize these two indicators were the best. First, the value of the learning rate was fixed at 0.001 and the optimal values of the other parameters were searched. Second, the value of the correction coefficient was varied between 1, 0.1, 0.01, 0.001, and 0.0001 to find the optimal value, which was 0.01 in our experiment (see Figure 3). Third, we set the number of hidden layer nodes at 3, 5, 10, 30, 50, 70, or 90 to find the best parameter, which was 5. The results are shown in Table 2. Fourth, the optimal number was found by changing the number of hidden layers between 1, 2, 3, and 4; the optimal value was 1, as seen in Table 3. Fifth, after the above parameters were determined, the optimal prediction performance could be obtained by

selecting the sigmoid activation function in Table 4. Finally, the learning rate was changed between 1, 0.1, 0.01, 0.001, 0.0001, 0.00001, and 0.000001 to find the optimal rate. When the learning rate changed from 0.001 to 0.000001, the values of the evaluation indicators were basically unchanged. Therefore, we set the learning rate to 0.001 (see Figure 4).

Figure 3. The results of the AE-SPLMLP model obtained with different correction coefficient values.

Table 2. The results of the AE-SPLMLP model obtained with different numbers of hidden layer neurons.

	MSE	MAE
3	0.00602	0.05149
5	0.00578	0.04752
10	0.00655	0.05432
30	0.00747	0.05947
50	0.00594	0.04734
70	0.00759	0.05913
90	0.00612	0.04758

Table 3. The results of the AE-SPLMLP model obtained with different numbers of hidden layers.

	MSE	MAE
1	0.00554	0.05121
2	0.00798	0.05921
3	0.00682	0.05109
4	0.00745	0.05808

Table 4. The results of the AE-SPLMLP model obtained with different activation functions.

	MSE	MAE
Sigmoid	0.00600	0.04979
ReLu	0.00701	0.05171
Tanh	0.00913	0.05601

Figure 4. The results of the AE-SPLMLP model obtained with different learning rate values.

3.3. The Results Obtained on Several Datasets

This study chose the SVR [32], GBDT [33], XgBoost [34], and LightGBM [35] models based on ML and an MLP and an LSTM model based on deep learning as the baseline models. To further prove that the denoising ability of the sparse deep autoencoder and SPL approach could effectively solve the problem of the model falling into local optimal solutions, this study introduced two variant models: AE-MLP and SPLMLP. The SPLMLP model removed the sparse deep autoencoder-based denoising module so that we could further evaluate the effect of the SPL framework. The AE-MLP model removed the SPL module so that we could further evaluate the effect of the sparse deep autoencoder.

From Table 5, the average MSE, RMSE CV-RMSE, and MAE values of the AE-SPLMLP model were 0.00614, 3.8×10^{-5} , 5.9×10^{-5} , and 0.05087, respectively. The evaluation metrics of the AE-SPLMLP model were 71.07%, 94.72%, 94.64%, and 54.19% better than those of the SVR model; 19.84%, 36.98%, 37.17%, and 11.41% better than those of the GBDT model; 22.18%, 40.25%, 38.73%, and 12.22% better than those of the XgBoost model; 15.89%, 29.50%, 18.51%, and 6.88% better than those of the LightGBM model; 51.04%, 76.25%, 60.67%, and 36.52% better than those of the MLP model; 93.01%, 99.51%, 99.66%, and 80.40% better than those of the LSTM model; and 24.10%, 42.25%, 35.59%, and 21.73% better than those of the AE-MLP model, respectively. The prediction performance of the AE-MLP model was significantly better than that of the MLP model, which demonstrated that adding a sparse deep autoencoder to the model could effectively solve the problem of noise in the multiscale power load data. In addition, the experiment showed that the SPLMLP model was also significantly better than the MLP model, which proved that incorporating the SPL method into the model could effectively improve its prediction performance. Furthermore, the AE-SPLMLP model achieved the best fitting ability on the testing set, as shown in Figure 5. Additionally, the prediction performance of the AE-SPLMLP model was very stable for the five random experiments shown in Table 6. All these results demonstrate the necessity of multiscale fusion and that the AE-SPLMLP model was significantly better than the existing STLF prediction models. At the same time, the AE-SPLMLP model obtained relative performance gains of 24.10% in terms of the MSE, 35.59% in terms of the CV-RMSE, and 21.73% in terms of the MAE over the AE-MLP model. Moreover, the convergence speed of the AE-SPLMLP model was faster than that of the AE-MLP model, as shown in Figure 6. These results further indicate that adding the SPL framework to the model could

effectively solve the problem of the model easily falling into local optimal solutions and improve the prediction performance of the model.

Table 5. Average results obtained by all models in five randomized experiments.

Model	MSE	RMSE	CV-RMSE	MAE
SVR	0.02123 ± 0.0165	0.00072 ± 0.0011	0.00110 ± 0.0017	0.11105 ± 0.0490
GBDT	0.00766 ± 0.0013	$6.03 \times 10^{-5} \pm 1.92 \times 10^{-5}$	$9.39 \times 10^{-5} \pm 3.12 \times 10^{-5}$	0.05742 ± 0.0075
XgBoost	0.00789 ± 0.0012	$6.36e \times 10^{-5} \pm 1.91 \times 10^{-5}$	$9.63 \times 10^{-5} \pm 2.72 \times 10^{-5}$	0.05795 ± 0.0089
LightGBM	0.00730 ± 0.0008	$5.39e \times 10^{-5} \pm 1.16 \times 10^{-5}$	$7.24 \times 10^{-5} \pm 1.70 \times 10^{-5}$	0.05463 ± 0.0060
MLP	0.01254 ± 0.0023	$0.00016 \pm 5.75 \times 10^{-5}$	$0.00015 \pm 8.33 \times 10^{-5}$	0.08014 ± 0.0094
LSTM	0.08782 ± 0.0005	$0.00771 \pm 5.10 \times 10^{-5}$	0.01716 ± 0.0113	0.25955 ± 0.0051
AE-MLP	0.00809 ± 0.0006	$6.58 \times 10^{-5} \pm 9.08 \times 10^{-5}$	$9.16 \times 10^{-5} \pm 4.00 \times 10^{-5}$	0.06499 ± 0.0035
SPLMLP	0.00834 ± 0.0006	$6.95 \times 10^{-5} \pm 1.14 \times 10^{-5}$	$9.71 \times 10^{-5} \pm 1.69 \times 10^{-5}$	0.06365 ± 0.0031
AE-SPLMLP	0.00614 ± 0.0005	$3.8 \times 10^{-5} \pm 6.69 \times 10^{-6}$	$5.9 \times 10^{-5} \pm 9.81 \times 10^{-6}$	0.05087 ± 0.0043

Figure 5. Prediction performance of the proposed AE-SPLMLP model on the testing data. The x -axis represents days; the y -axis represents the standardized electricity load values for the testing data.

Figure 6. The convergence curve results of different models: (a) the convergence curve result of the AE-MLP model on the power load dataset, (b) the convergence curve result of the AE-SPLMLP model on the power load dataset.

Table 6. Average results obtained by AE-SPLMLP model in five randomized experiments.

Testing Set	MSE	RMSE	CV-RMSE	MAE
1	0.00578	3.34×10^{-5}	5.25×10^{-5}	0.04752
2	0.00586	3.43×10^{-5}	5.51×10^{-5}	0.04938
3	0.00558	3.12×10^{-5}	4.77×10^{-5}	0.04597
4	0.00649	4.21×10^{-5}	6.45×10^{-5}	0.05391
5	0.00701	4.92×10^{-5}	7.53×10^{-5}	0.05759
Average	0.00614 ± 0.0005	$3.8 \times 10^{-5} \pm 6.69 \times 10^{-6}$	$5.9 \times 10^{-5} \pm 9.81 \times 10^{-6}$	0.05087 ± 0.0043

4. Discussion and Conclusions

In this study we designed AE-SPLMLP, a new STLF prediction model based on a sparse deep autoencoder and the strategy of combining SPL and an MLP (SPLMLP). The sparse deep autoencoder model was used to solve the noise problem caused by the fusion of multiscale power load data. The SPL method solved the local optimal solution problem and sped up the model convergence. The idea of SPL is to adopt a strategy that operates from simple to complex to gradually incorporate samples into the model. In theory, Meng et al. [36] proved that the AOS algorithm utilized to solve the SPL problem is identical to a majorization–minimization (MM) algorithm implemented on a latent SPL objective function, and they used existing MM knowledge to prove the convergence and stability of the SPL solution strategy. Our experimental results also show that the AE-SPLMLP model was 6.88–99.66% better than the existing models. In addition, to further prove the effectiveness of autoencoder denoising and the fact that the SPL framework can alleviate the local optimal solution problem, this paper also compared the AE-SPLMLP model with the AE-MLP and SPLMLP models. Among these, the prediction performance of the AE-SPLMLP model was significantly better than that of the SPLMLP model, which showed that the sparse autoencoder could fuse multiscale data well and could effectively solve the noise problem observed after data fusion. Similarly, the comparison results between the AE-SPLMLP and AE-MLP models indicated that, compared with the traditional learning curve method, incorporating SPL into the AE-MLP model could effectively alleviate the problem of the model falling into local optimal solutions, speed up the model convergence, and improve the prediction performance of the model. In conclusion, the AE-SPLMLP model can serve as a powerful tool to forecast short-term multiscale power loads. However, the AE-SPLMLP model has some limitations; for example, the SPL framework has a high time complexity, which leads to slow model training. Improving the AE-SPLMLP model to speed up its training process is a problem that needs to be addressed in the future.

Author Contributions: Conceptualization, M.L. and X.X.; methodology, M.L.; software, M.L.; validation, M.L., X.X. and D.Z.; formal analysis, M.L.; investigation, M.L.; resources, M.L.; data curation, X.X.; writing—original draft preparation, M.L.; writing—review and editing, D.Z.; visualization, X.X.; supervision, D.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Macau Science and Technology Development Funds (grants 0158/2019/A3 and 0056/2020/AFJ) from the Macau Special Administrative Region of the People’s Republic of China.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: We deeply acknowledge Macau University of Science and Technology for supporting this study through Macau Science and Technology Development Funds (grants 0158/2019/A3 and 0056/2020/AFJ) from the Macau Special Administrative Region of the People’s Republic of China.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Valor, E.; Meneu, V.; Caselles, V. Daily air temperature and electricity load in Spain. *J. Appl. Meteorol.* **2001**, *40*, 1413–1421. [CrossRef]
2. Soaresm, L.J.; Medeiros, M.C. Modeling and forecasting short-term electricity load: A comparison of methods with an application to Brazilian data. *Int. J. Forecast.* **2008**, *24*, 630–644. [CrossRef]
3. Paatero, J.V.; Lund, P.D. A model for generating household electricity load profiles. *Int. J. Energy Res.* **2006**, *30*, 273–290. [CrossRef]
4. Hong, T.; Wilson, J.; Xie, J. Long term probabilistic load forecasting and normalization with hourly information. *IEEE Trans. Smart Grid* **2013**, *5*, 456–462. [CrossRef]
5. Gavrilas, M.; Ciutea, I.; Tanasa, C. Medium-term load forecasting with artificial neural network models. In Proceedings of the 16th International Conference and Exhibition on Electricity Distribution, 2001. Part 1: Contributions, CIRED, Amsterdam, The Netherlands, 18–21 June 2001.
6. Yalcinoz, T.; Eminoglu, U. Short term and medium term power distribution load forecasting by neural networks. *Energy Convers. Manag.* **2005**, *46*, 1393–1405. [CrossRef]
7. Moghram, I.; Rahman, S. Analysis and evaluation of five short-term load forecasting techniques. *IEEE Trans. Power Syst.* **1989**, *4*, 1484–1491. [CrossRef]
8. Mandal, P.; Senjyu, T.; Funabashi, T. Neural networks approach to forecast several hour ahead electricity prices and loads in deregulated market. *Energy Convers. Manag.* **2006**, *47*, 2128–2142. [CrossRef]
9. Kumar, A.; Luthra, S.; Mangla, S.K.; Kazançoğlu, Y. COVID-19 impact on sustainable production and operations management. *Sustain. Oper. Comput.* **2020**, *1*, 1–7. [CrossRef]
10. Zhong, H.; Tan, Z.; He, Y.; Xie, L.; Kang, C. Implications of COVID-19 for the electricity industry: A comprehensive review. *CSEE J. Power Energy Syst.* **2020**, *6*, 489–495.
11. Gupta, M.; Abdelmaksoud, A.; Jafferany, M.; Lotti, T.; Sadoughifar, R.; Goldust, M. COVID-19 and economy. *Dermatol. Ther.* **2020**, *33*, e13329. [CrossRef]
12. Peng, T.; Hubele, N.; Karady, G. Advancement in the application of neural networks for short-term load forecasting. *IEEE Trans. Power Syst.* **1992**, *7*, 250–257. [CrossRef]
13. Solyali, D. A comparative analysis of machine learning approaches for short-/long-term electricity load forecasting in Cyprus. *Sustainability* **2020**, *12*, 3612. [CrossRef]
14. Li, Y.-C.; Fang, T.-J.; Yu, E.-K. Study of support vector machines for short-term load forecasting. *Proc. CSEE* **2003**, *23*, 55–59.
15. Barman, M.; Choudhury, N.D.; Sutradhar, S. A regional hybrid GOA-SVM model based on similar day approach for short-term load forecasting in Assam, India. *Energy* **2018**, *145*, 710–720. [CrossRef]
16. Barman, M.; Choudhury, N.B.D. Season specific approach for short-term load forecasting based on hybrid FA-SVM and similarity concept. *Energy* **2019**, *174*, 886–896. [CrossRef]
17. Liu, Q.; Shen, Y.; Wu, L.; Li, J.; Zhuang, L.; Wang, S. A hybrid FCW-EMD and KF-BA-SVM based model for short-term load forecasting. *CSEE J. Power Energy Syst.* **2018**, *4*, 226–237. [CrossRef]
18. Askari, M.; Keynia, F. Mid-term electricity load forecasting by a new composite method based on optimal learning MLP algorithm. *IET Gener. Transm. Distrib.* **2020**, *14*, 845–852. [CrossRef]
19. Tang, J.; Zhao, J.; Zou, H.; Ma, G.; Wu, J.; Jiang, X.; Zhang, H. Bus Load Forecasting Method of Power System Based on VMD and Bi-LSTM. *Sustainability* **2021**, *13*, 10526. [CrossRef]
20. Butt, F.M.; Hussain, L.; Mahmood, A.; Lone, K.J. Artificial Intelligence based accurately load forecasting system to forecast short and medium-term load demands. *Math. Biosci. Eng.* **2020**, *18*, 400–425. [CrossRef]
21. Kong, W.; Dong, Z.Y.; Jia, Y.; Hill, D.J.; Xu, Y.; Zhang, Y. Short-term residential load forecasting based on LSTM recurrent neural network. *IEEE Trans. Smart Grid* **2017**, *10*, 841–851. [CrossRef]
22. Mujeeb, S.; Javaid, N.; Ilahi, M.; Wadud, Z.; Ishmanov, F.; Afzal, M.K. Deep long short-term memory: A new price and load forecasting scheme for big data in smart cities. *Sustainability* **2019**, *11*, 987. [CrossRef]
23. Kontogiannis, D.; Bargiotas, D.; Daskalopulu, A. Minutely active power forecasting models using neural networks. *Sustainability* **2020**, *12*, 3177. [CrossRef]
24. Mu, X.-Y.; Zhang, T.-Y.; Zhou, Y. Short-term load forecasting on multi-scale Gaussian model. *Microelectron. Comput.* **2008**, *12*. Available online: https://en.cnki.com.cn/Article_en/CJFDTotal-WXYJ200812056.htm (accessed on 24 May 2021).
25. Ng, A. Sparse autoencoder. *CS294A Lect. Notes* **2011**, *72*, 1–19.
26. Kumar, M.; Packer, B.; Koller, D. Self-paced learning for latent variable models. *Adv. Neural Inf. Process. Syst.* **2010**, *23*, 1189–1197.
27. Jiang, L.; Meng, D.; Yu, S.-I.; Lan, Z.; Shan, S.; Hauptmann, A. Self-paced learning with diversity. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 2078–2086.
28. Wi, Y.-M.; Joo, S.-K.; Song, K.-B. Holiday load forecasting using fuzzy polynomial regression with weather feature selection and adjustment. *IEEE Trans. Power Syst.* **2011**, *27*, 596–603. [CrossRef]
29. Haq, M.R.; Ni, Z. A new hybrid model for short-term electricity load forecasting. *IEEE Access* **2019**, *7*, 125413–125423. [CrossRef]
30. Chen, K.; Chen, K.; Wang, Q.; He, Z.; Hu, J.; He, J. Short-term load forecasting with deep residual networks. *IEEE Trans. Smart Grid* **2018**, *10*, 3943–3952. [CrossRef]
31. Zeng, P.; Jin, M.; Elahe, M.F. Short-term power load forecasting based on cross multi-model and second decision mechanism. *IEEE Access* **2020**, *8*, 184061–184072. [CrossRef]

32. Tan, Z.; Zhang, J.; He, Y.; Zhang, Y.; Xiong, G.; Liu, Y. Short-term load forecasting based on integration of SVR and stacking. *IEEE Access* **2020**, *8*, 227719–227728. [[CrossRef](#)]
33. Liu, S.; Cui, Y.; Ma, Y.; Liu, P. Short-term load forecasting based on GBDT combinatorial optimization. In Proceedings of the 2nd IEEE Conference on Energy Internet and Energy System Integration (EI2), Beijing, China, 20–22 October 2018; pp. 1–5.
34. Zheng, H.; Yuan, J.; Chen, L. Short-term load forecasting using EMD-LSTM neural networks with a Xgboost algorithm for feature importance evaluation. *Energies* **2017**, *10*, 1168. [[CrossRef](#)]
35. Wang, Y.; Chen, J.; Chen, X.; Zeng, X.; Kong, Y.; Sun, S.; Guo, Y.; Liu, Y. Short-term load forecasting for industrial customers based on TCN-LightGBM. *IEEE Trans. Power Syst.* **2020**, *36*, 1984–1997. [[CrossRef](#)]
36. Zhao, Q.; Meng, D.; Jiang, L.; Xie, Q.; Xu, Z.; Hauptmann, A.G. Self-paced learning for matrix factorization. In Proceedings of the 29th AAAI Conference on Artificial Intelligence, Austin, TX, USA, 25–30 January 2015.