International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

Real-Time Efficient Short-Term Peak Load and Day-Ahead Electricity Load Forecasting System Using Machine Learning Approach

S. Vasudevan[1*], Dr. K. Jothinathan[2]

Submitted: 26/01/2024 **Revised**: 04/03/2024 **Accepted**: 12/03/2024

Abstract: Short-term load forecasting is crucial for efficiently managing electricity usage, spanning from weekly down to sub-hourly intervals. This practice not only saves resources but also ensures customer needs are met promptly. Day-ahead peak demand forecasting plays a vital role in load management, aiding in power system planning and operation. Yet, due to its complex non-linear nature, accurately predicting peak loads presents significant challenges. Therefore, this research proposed a hybrid predictive deep learning with an optimization algorithm to forecast precise electricity for 30-minute intervals. First, the maximum and minimum ranges of various parameters are determined by looking at historical data. The real-time datasets from January 1 to December 31st 2021 were used to derive hourly real-time system demand load data. The original dataset undergoes preprocessing to reconstruct its electrical characteristics. Zero-mean normalization is applied to both load and temperature data to standardize them. In intricate electric load systems, redundant information can hinder accurate pattern extraction for load forecasting. Principal Component Analysis Network (PCANet) identifies relevant features while eliminating redundancy. A DeepWalk Gated Recurrent Unit Model (DWGRU) framework is then constructed to capture temporal dependencies from historical sequences, integrating spatial, temporal, and semantic features for load forecasting. These extracted features are dynamically combined using an attention mechanism. Subsequently, a Hybrid Sampling and Self Attention with Deep Neural Network (HSSA-DNN) is employed to forecast 30-minute peak loads efficiently, utilizing Improved Moth Flame Optimization (IMFO). The proposed methodology is implemented using Matlab Simulink software. Forecasting accuracy is assessed using statistical error metrics such as Mean Absolute Percentage Error (MAPE), Mean Absolute Error (MAE), and Root Mean Square Error (RMSE) to identify optimal models. Experimental results showcase the superior accuracy of the proposed approach, evidenced by precise direction, equality, stability, correlation, comprehensive accuracy, and statistical performance analysis. Comparisons with existing methods reveal a reduction in Mean Absolute Scaled Error of up to 0.2088. Achieving a high day-ahead net load forecasting accuracy of 99.15% underscores the effectiveness of load forecasting. This highlights the critical role of input data structure and quality in further enhancing forecasting accuracy and reliability.

Keywords: *Short-Term Peak Load, Real-Time System, Zero-Mean Normalization, Principal Component Analysis Network, Electricity Load Forecasting System, Sampling and Self Attention, Improved Moth Flame Optimization*

1. Introduction

An electric power system, also referred to as an electric grid, encompasses a collection of electrical components that distribute, transmit, and utilize power for residential and industrial purposes [1]. Traditional power systems typically produce electricity in centralized facilities such as power plants. These plants generate power, which then travels across extensive transmission lines to radial distribution networks. Along this journey, voltage is decreased, ultimately reaching consumers for utilization. [2]. Traditional power systems face challenges like network reliability, robustness, and energy price reduction. Distributed generations have been rapidly increased to address these issues [3-4]. The integration of distributed generation alongside advanced communication technologies has led to the emergence of smart grids [5]. A key focus within smart grid development revolves around devising novel strategies for demand response, aiming to maximize efficiency during peak periods without necessitating an increase in production capacity [6].

These networks are required to make critical operational decisions based on electricity consumption levels. Long short-term memory (LSTM) plays a vital role in ensuring the effective management of such networks [7].

Short-term load forecasting (STLF) plays a crucial role in enabling networks to make numerous operational decisions effectively. The behavior of load time series tends to be intricate [8], thus necessitating the use of artificial intelligence (AI) models capable of handling nonlinear functions for electricity load forecasting. Commonly employed AI models include artificial neural networks (ANN), support vector machines (SVM), bagged regression trees, and random forests. Over time, numerous ANN-based prediction models,

[1]*Research Scholar, Department of Electrical Engineering, Annamalai University, Annamalai Nagar, Chidambaram, 608 002, Tamil Nadu, India, Email: vasudevanaetneb@gmail.com*
[2]*Associate Professor, Department of Electrical Engineering, Annamalai University, Annamalai Nagar, Chidambaram, 608 002, Tamil Nadu, India, Email: jothi.eeau@yahoo.com*

such as extreme learning machines (ELM), have been proposed [9]. A diverse range of AI algorithms, including Deep Learning and Neuro-Fuzzy methods, have been developed and applied in the electrical systems domain to optimize power system operation and management, as well as for load and electricity price forecasting [10]. AI-based techniques are increasingly being adopted for load forecasting in smart grids, distributed systems, and next-day demand forecasting. Additional AI-based approaches encompass fuzzy logistic techniques, specialized network structures, Bayesian neural systems, and support vector machines [11]. Despite extensive research efforts, achieving error-free and precise STLF remains challenging due to the non-stationary nature of load data and the persistent dependencies associated with forecasting horizons [12]. This research aims to investigate short-term load forecasting in electrical power systems using AI techniques. Unlike other AI methods that rely on load statistics, the study utilizes Long Short-Term Memory (LSTM) for forecasting over long time horizons. The study is structured into sections covering literature review, problem definition and motivation, proposed methodology, experimentation and result discussion, and research conclusions.

2. Literature Survey

Panagiotou *et al* [13] A novel integrated model was proposed for short-term electricity load forecasting. The findings demonstrate that the accuracy of this dynamic integrated model reaches up to 99% Hu et al [14]. Another study Rao et al. [15] examined a dynamic Artificial Neural Network (ANN) model called Meta-ANN, specifically developed for short-term grid load forecasting. Results from numerical analysis indicate that Meta-ANN achieves superior accuracy and robustness by effectively capturing nonstationary patterns in grid loads. In a separate investigatio,a load forecasting method was developed for cluster microgrids employing machine learning algorithms such as linear regression, support vector machines, and ANN, with the Levenberg-Marquardt optimization algorithm yielding the most favorable outcomes. Furthermore, a study Bashir et al [16] assessed the performance of a hybrid technique by evaluating metrics like MAPE, RMSE, and Mean Average Error (MAE). Results affirm that the proposed hybrid models surpass standalone approaches such as Autoregressive Integrated Moving Average (ARIMA), LSTM, and Prophet Model, showcasing reduced errors with minimal computation time. Tang et al. [17] They suggested a Short-Term Load Forecasting (STLF) model based on Temporal Convolutional Network (TCN), illustrating its efficacy in enhancing both forecast accuracy and generalization capacity.

Liu *et al* [18] They introduced a hybrid method for short-term building load probability density forecasting, which combines Orthogonal Maximum Correlation Coefficient (OMCC) feature selection with Convolutional Gated Recurrent Unit (CGRU) quantile regression. Results from simulations conducted across three different buildings confirm the dependability of the proposed model for short-term building-level probabilistic load forecasting tasks. Ghenai *et al* [19] They constructed a predictive model aimed at building energy planning, with the objective of balancing the supply from renewable power systems with the building's electrical load demand. Key findings indicate that the predictive model exhibits exceptional accuracy in forecasting a building's energy consumption. Meng *et al* [20] Explored a short-term load forecasting method that incorporates empirical mode decomposition (EMD), bidirectional long short-term memory (BiLSTM), and an attention mechanism. The experiments indicated that the optimal number of Intrinsic Mode Functions (IMFs) recommended for this approach is either three or four, considering both prediction accuracy and computational efficiency. Matrenin *et al* [21] Executed the method for identifying the most impactful features. Analysis of the total forecast error has demonstrated that the attributes of the proposed models exhibit high quality and precision, thereby making them suitable for accurately forecasting the actual load of a power system. Alrasheedi *et al* [22] Suggested hybrid deep learning (DL) techniques to improve outcomes in load forecasting for Saudi smart grids. The forecasted results showcase the efficacy of the proposed hybrid DL models. Specifically, CNN-GRU and CNN-RNN achieved improvements of 1.4673% and 1.222% in load forecasting accuracy measured by NRMSE, respectively. This research significantly contributes to short-term load forecasting in electrical power systems through the utilization of hybrid machine learning algorithms.

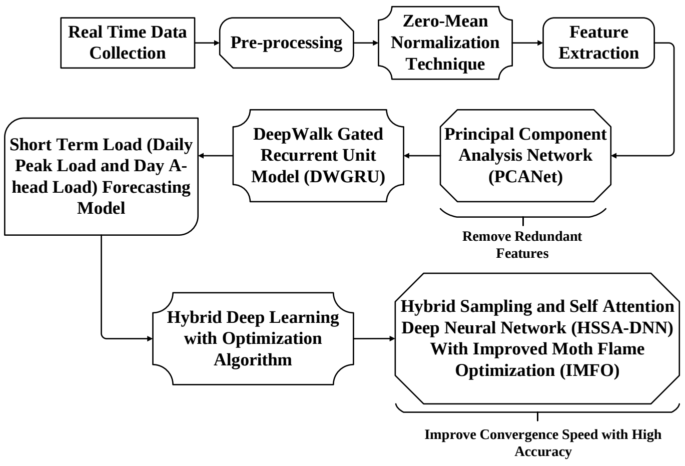
3. Research Problem Definition And Motivation

Electric power companies are responsible for providing high-quality electricity to consumers safely and efficiently. They must plan, manage, and operate the system while addressing economic and technological challenges. Accurate assessment of current and projected demand is crucial for optimal planning and operation. Electric load forecasting helps predict the amount of electricity needed to meet demand. Load forecasting can be categorized into three main types: short-term, mid-term, and long-term. Short-term forecasting stands out for its widespread usage, especially for facilitating day-to-day operations.

The electrical system's infrastructure has undergone significant changes, impacting customer consumption habits. The study seeks to create a hybrid model to precisely forecast load consumption in power networks, aiming to rectify deficiencies observed in current AI-based systems such as linear segments, complexity, and convergence challenges. It particularly focuses on forecasting peak load demand and day-ahead load for 30-minute intervals.

4. Proposed Research Methodology

Demand forecasting is crucial for power management and decision-making due to technological challenges and rising electrical load. Time series models like linear regression struggle with complex nonlinear load time series. AI models, deep learning, and machine learning offer reliable, quick, and efficient solutions. A hybrid approach incorporating deep learning techniques optimizes parameters for short-term load forecasting.



Real Time Data Collection
Pre-processing
Zero-Mean Normalization Technique
Feature Extraction
Short Term Load (Daily Peak Load and Day A-head Load) Forecasting Model
DeepWalk Gated Recurrent Unit Model (DWGRU)
Principal Component Analysis Network (PCANet)
Remove Redundant Features
Hybrid Deep Learning with Optimization Algorithm
Hybrid Sampling and Self Attention Deep Neural Network (HSSA-DNN) With Improved Moth Flame Optimization (IMFO)
Improve Convergence Speed with High Accuracy

Fig. 1: Block Diagram of the Proposed Work

The proposed methodology utilizes real-time data from Tamilnadu, which is pre-processed to enhance the learning of the input-output relationship, as depicted in Figure 1. During pre-processing, mathematical operations like normalization, ranking, and correlation are applied. The process utilizes zero-mean normalization, principal component analysis network, and Deep Walk Gated Recurrent Unit Model, and presents a STLF model using hybrid deep learning and optimization algorithms for improved convergence speed and accuracy.

4.1. Data Collection

The dataset includes year, month, date, day, hour, weekday, working/holiday, dew point, dry bulb temperatures, and hourly load. The Southern region power grid website, the TNEB load dispatch centre website, and official documents were used to gather real-time data from the Tamil Nadu districts of Madurai and Chennai. The data covered one year. The dataset included data for each hour from January 1, 2021, to December 31, 2021, including power load, temperatures, humidity, wind speed, day type, and other variables. The weather data, time indicators, and load demand statistics are all part of the first dataset-gathering process. Tests are administered to each one-year group. The data are also combined with the load data from the previous week (as a redundant input) and the most recent day (24 hours), likewise as a redundant input.

4.2. Zero-Mean Normalization Technique for Data Pre-Processing

The study uses a zero-mean normalization technique for data normalization on load and temperature variables, enhancing numerical stability in NN training. The Z-norm method uses mean and variance estimates for distribution scaling, allowing off-line parameter calculation during training. Log-likelihood scores are used to estimate the mean and variance specific to each speaker for the imposter distribution, following the testing of a speaker model against sample impostor utterances. The form of normalisation exists in Equation (1).

$$S = \frac{\log\big(P(m|O)\big) - \mu_I}{\sigma_I} \qquad (1)$$

Where, S is the distribution-normalized score μ_I and $\sigma_I$are the estimated impostor parameters for speaker model m.

Different normalisation methods can be categorized based on their intended application, such as distribution scaling or score normalisation. Distribution scaling can be speaker-centric or impostor-centric, with two primary methods: Bayesian world model implementation and cohort normalisation. Cohort normalisation compares speaker models to unconstrained cohorts, using similarity metrics. The size of the cohort influences normalisation behavior, with larger groups exhibiting impostor-centric behavior.

Cohort normalisation becomes more speaker-centric as the number of speakers decreases. For text-dependent verification, a cohort size of less than five yields the best results.

4.3. Feature Extraction and Selection

The study investigates the influence of multiple variables on load parameters in electric power load series, highlighting the potential for load signal instability. To enhance precision, the research converts load data into time-series data using the Principal Component Analysis Network and Deep Walk Gated Recurrent Unit Model. The system involves tagging and embedding procedures.

4.3.1. Principal Component Analysis Network (PCANet)

The feature extraction processes utilize the Principal Component Analysis Network (PCANet) model to provide high-dimensional features. The PCANet architecture represents a streamlined and faster variant of CNN designed for image classification. It employs cascaded principal component analysis (PCA), binary hashing, and blockwise histograms across three layers. Its key components are PCA filters, which extract the highest energy eigenvectors from input training data, and convolution filters, which capture the primary variance of the input.

Input Layer: The preceding phase produced N input data of size $m \times n$, allowing us to assume that the filter size is $\{I_i\}_{i=1}^{N}$ throughout. Equation (2) illustrates the matrix representation of the *i-th* dataset.

$$I_i = \begin{bmatrix} i_{11} & i_{12} & \cdots & i_{1n} \\ i_{21} & i_{22} & \cdots & i_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ i_{m1} & i_{m2} & \cdots & i_{mn} \end{bmatrix} \qquad (2)$$

Wherein, m and n represent the width and height dimensions of the data, respectively.

Hidden Layer: This layer consists of two stages. In the first stage, the border of Ii is zero-padded, then convolved with the $k_1 \times k_2$ filter to extract patches of the *i-th* data. This process enables the determination of local features within the filter patch in a single step. To achieve $x_{i,1}, x_{i,2}, \dots, x_{i,mn} \in \mathbb{R}^{k_1k_2}$ where $\bar{x}_{i,j}$a mean-removed patch is, the developers must first remove the patch mean from each patch. The identity matrix is established by uniformly transforming each input data, and this method yields X (Equation (3)).

$$X = [\bar{X}_1, \bar{X}_2, \dots, \bar{X}_N] \in \mathbb{R}^{k_1k_2 \times Nmn} \qquad (3)$$

The PCA filters are gathered following the PCA technique and may be expressed as Equation (4).

$$W_l^1 = matk_1, k_2\big(q_l(XX^T)\big) \in \mathbb{R}^{k_1k_2}, l = 1,2,\dots,L_1 \quad (4)$$

In this context, L_i denotes the number of filters, and q_l (XX^T) selects the *l-th* primary eigenvector of XX^T. *matk*$_1$, *k*$_2$ *(v)* represents a function that converts column vectors to matrices, while W_l^1 corresponds to the *l-th* PCA filter used to extract high-dimensional features in the first stage. Convoluting the $l-th$ PCA filter with the input data provides the first stage's $l-th$ filter output.

$$I_i^l = I_i * W_l^1, \quad i = 1,2,\dots,N \qquad (5)$$

While it is zero-padded before convolution with W_l^1, I_i^lalso has the same size as I_i. The process is reiterated similarly to the first stage: aggregating all patches of I_i^l (Equation (5)), subtracting the mean of each patch, and then deriving $\bar{Y}_i^l = [\bar{y}_{i,l,1}, \bar{y}_{i,l,2}, \dots, \bar{y}_{i,l,mn}] \in \mathbb{R}^{k_1k_2 \times mn}$, where $\bar{y}_{i,l,j}$ $Y^l = [\bar{Y}_1^l, \bar{Y}_2^l, \dots, \bar{Y}_N^l] \in \mathbb{R}^{k_1k_2 \times Nmn}$ for the matrix, and amalgamating the output of all lth filters to yield the following Equation (6).

$$Y = [Y^1, Y^2, .., Y^{L_1}] \in \mathbb{R}^{k_1k_2 \times L_1Nmn} \qquad (6)$$

The next step is utilising the PCA technique to obtain the second-stage PCA filters illustrates Equation (7).

$$W_l^2 = matk_1, k_2\big(q_l(YY^T)\big) \in \mathbb{R}^{k_1,k_2}, \quad l = 1,2,\dots L_2 \quad (7)$$

According to the Equation (8), each input I_i^l of the second stage will produce L_2 the data size of $m \times n$,

$$O_i^l = \{I_i^l * W_l^2\}_{l=1}^{L2} \quad (8)$$

Where O_i^l represents the output of the $i-th$ image, and L_1L_2 ignifies the number of output images.

Output Layer: The results from the second stage, $\{I_i^l * W_l^2\}_{l=2}^{L2}$ will be binarized in this layer before being transformed into decimal matrices as follows Equation (9) and (10).

$$\Gamma_i^l = \sum_{l=1}^{L_2} 2^{l-1} H\big(I_i^l * W_l^2\big) \quad (9)$$

$$f_i = [Bhist(\Gamma_i^l), \dots, Bhist(\Gamma_i^{L_1})]^T \in \mathbb{R}^{(2^{L2})L_1B} \quad (10)$$

The $i-th$ data's output is a decimal matrix, and H (·) represents a Heaviside step function, with a value of one for positive entries and zero otherwise. After this encoding process, the input image Ii is converted into a set of block-wise histograms for eachΓ_i^l, $l = 1, \dots L_1$. This is achieved by calculating histograms for each of these blocks and then combining all of the B histograms into a single vector, denoted as *Bhist* (Γ_i^l).

4.3.2. Deep Walk Gated Recurrent Unit Model

In this phase, they provide further detail about the ST-DWGU ensemble deep learning framework's architecture. The ST-DWGRU comprises multiple layers, each consisting of three elements: the extraction of spatial and temporal features, and the extraction of semantic features. The outcomes of the prediction are output by the final prediction component in Equation (11).

$$y_{t+T} = f\left(x_{t-h,\ldots,};\, m_{t-h,\ldots,t};\, A\right), \quad (11)$$

In contrast to the prior prediction model, each layer of ST-DWGRU also learns semantic and spatiotemporal features. GCNs cannot fully learn the position information of nodes since the majority of graph neural networks construct node embedding by combining data from each node's q-hop neighbourhood and are hence structure-aware. varied location crossings play varied roles in the road network, thus to describe the semantic information of various intersections, they require the position information that is encoded in the nodes. The suggested ST-DWGRU learns semantic and spatiotemporal features, which is more appropriate for the real case.

In Euclidean space, a CNN effectively captures spatial effects by considering the weights of neighboring pixels. However, CNNs face limitations in directly extracting spatial aspects from the structure of a road network, especially when the number of neighboring junctions and road segments varies. The urban road network is represented by an undirected graph $G = (V, E)$ where V represents the set of graph vertices and E represents the set of graph edges. Utilizing the input of the adjacency matrix of the road network, the graph convolution operation extracts the characteristics of the road network structure. A two-layer Graph Convolutional Network (GCN) can be described as Equation (12):

$$Z = f(X, A) = soft\max\left(\hat{A}ReLU\left(\hat{A}XW^{(0)}\right)W^{(1)}\right) \quad (12)$$

Apart from the input gate, output gate, and forget gate found in LSTM, GRU is a simplified version of an LSTM network with only two gates: an update gate and a reset gate. Let $X = (x_1, x_2, \ldots, x_t)$ be the input sequence, and input X to GRU to have GCN learn temporal properties beforehand. The following is a diagram of the precise calculation Equation (13).

$$s_t = GCN(x_t), \quad (13)$$

In this context, r_t represents the reset gate, z_t denotes the update gate, h_t signifies the current memory content, and h_t represents the current hidden state. Additionally, s_t denotes the output of the GCN at time *t*, h_t represents the hidden state at time *t*, x_t stands for the current input, h_{t-1} denotes the hidden state at the previous time step, and r_t indicates the reset gate. The weights and biases of the network are denoted by W and b respectively, the sigmoid activation function is represented by σ and the hyperbolic tangent activation function is denoted by tan_h. Thus, both temporal and spatial characteristics are present in the final GRU output.

Position-aware node embedding representation learning technique DeepWalk has lately gained much popularity. The method is composed of the random walk and update process components. The root node of a path is designated by the symbol W_{s_i}, and all other nodes are labelled as $\{W_{s_i}^1, W_{s_i}^2, W_{s_i}^3, \ldots, W_{s_i}^k\}$, where $W_{s_i}^k$ indicates the k-th intersection or segment along the path. The optimisation result and the associated vector representation is s_t^i generated via the Skip-gram technique (Equation (14)).

$$\hat{s} = f(W_i . S_i + b_i), \quad (14)$$

After obtaining the vector representation, this study generates the ultimate semantic information representation using a fully connected layer. In a truncated random walk, each random walk shares the same length, and upon traversing every junction or road segment, the random walk sequence matrix for the entire journey is obtained.

4.4. Short-Term Load Forecasting Model

The research suggests short-term peak load and day-ahead load forecasting through the utilization of the Hybrid Sampling and Self-Attention with Deep Neural Network (HSSA-DNN) and the Improved Moth Flame Optimization (IMFO) algorithm. The model predicts complex choice patterns, examines finite and infinite decision horizons, and uses transfer learning for spatial-temporal correlation. The IMFO algorithm accelerates convergence, improving load forecasts for power grid operations.

4.4.1. Hybrid Sampling and Self-Attention with Deep Neural Networks Mechanism

The research devised a deep learning model incorporating a sparse self-attention mechanism, which enhances model training by capturing data-specific features and improving model accuracy. The model consists of N layers, enhancing the recognition of hidden features.

Here, the approximating softmax by sampling a set S, a set of adjacent keys for each query created by the union of colliding keys using m hash tables. The estimator is computed using $|S|^{-1} \sum_{j \in S} \frac{p(Q_i, K_j)}{q(Q_i, K_j)} V_j$, where, Q_i is a query vector, K_j, V_j are key and value vectors in the sampling set S, and $p(Q_i, K_j)$ and $q(Q_i, K_j)$ are softmax probability and collision probability of given pairings. Due to the importance of sampling without replacement used in this technique, the samples become dependent on one another.

Initially, the extracted load data were inputted into the fully connected layer and softmax layer to train the self-attention score (Equation (15)).

$$a = softmax\big(g(l)\big) \quad (15)$$

Here, l represents the load binary vector, a signifies the attention score vector, and $g(\cdot)$ denotes the fully connected layer without activation. Specifically, $g(\cdot)$ serves as a linear operator, while $softmax\ (\cdot)$ functions as a non-linear activation operator. However, the sparse attention distribution is then computed as Equation (16):

$$\rho(e_{ij}) = p_{ij} = max\left\{0, \frac{e_{ij}-\tau(e_i)}{1-\lambda}\right\} \quad (16)$$

Where, $j \in \{1, \cdots L\}$ and τ is the threshold function. Thus, in order to emphasize the importance of critical features' influence on the sequence, the attention layer allocates the feature weight learned by the model to the input vector of the subsequent time step. Subsequently, the fully connected layer processes the final data. The estimated load value is determined after the fully linked layer's virtual function processing. The self-attention process unfolds as outlined below:

- Step 1: Compute the correlation between each current input feature and the present load.
- Step 2: Apply the Softmax formula to transform each correlation into a probabilistic form.
- Step 3: Multiply each resulting probability by the implicit representation of the corresponding input feature to signify the contribution of the feature to the predicted load. To forecast the subsequent load data, the contributions of all input features are then aggregated.

To denote the impact of each obtained probability on the anticipated load, multiply each outcome by the implicit representation of the corresponding input characteristic. Equations (17) - (19) can be used to describe the process.

$$e_t = Vtanh(Wh_t + b) \quad (17)$$

$$\alpha_t = \frac{exp(e_t)}{\sum_{j=1}^{n} exp(e_j)} \quad (18)$$

$$C_t = \sum_{t=1}^{n} a_t h_t \quad (19)$$

Where, the attention weight at time t and the weight score associated with certain aspects are e_t and α_t is The size of the input vector for the prediction model; V and W represent the weights of the multilayer perceptron used in calculating the attention weight; b stands for the bias parameter of the multilayer perceptron during attention weight calculation; and c denotes the output of the attention mechanism at time$t$. This research introduces attention mechanisms that leverage both past and future information characteristics, assigning different weights to input data to emphasize strong correlations and diminish less correlated components.

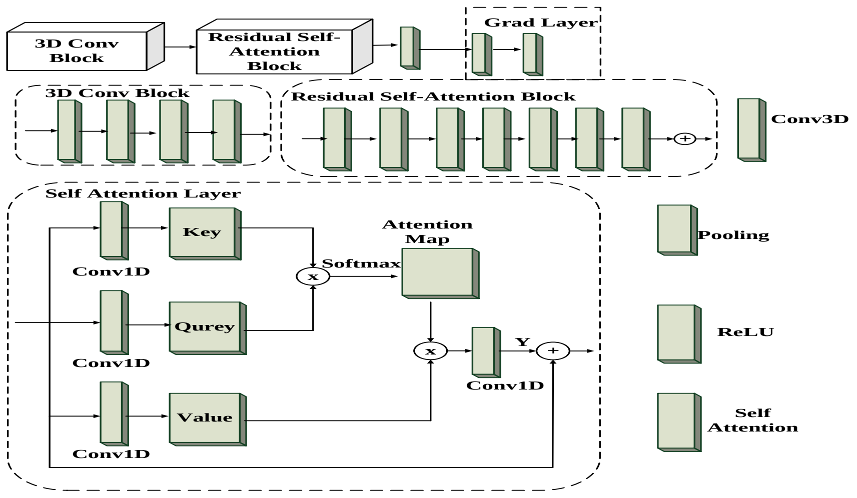


Fig. 2: Architecture of Self-Attention Deep Neural Network

Figure 2 presents a high-level conceptual framework featuring 3D convolution blocks, residual self-attention blocks, and explainable blocks. Conventional load fluctuation is primarily due to user energy use changes. Techniques like time series prediction and linear extrapolation can provide short-term forecasting precision. Due to the widespread availability of distributed energy sources, Equation (20) can be used to express the net load.

The actual net load is denoted by p_t, the user's electrical load is denoted by p', and power generation is indicated by p_t^g.

$$p_t = p'_t + p_t^g \quad (20)$$

The power generation is considered a load due to its erratic nature and differs from traditional power supplies. The prediction layer comprises three interconnected layers and a self-attention layer, which enriches load exploration. A solitary neuron utilizing a sigmoid activation function transforms input from the preceding layer into an output score. The formula is represented as Equation (21):

$$P(P1, P2) = s\big(Dens(F)\big) \tag{21}$$

Here, s represents a dense layer comprising one unit activated by the sigmoid function. As a result, load uncertainty rises, the fluctuation range widens, and on sunny days, power reversals will happen around noon. The usual forecasting approach will result in greater inaccuracies and be unable to predict the load preciselyThe study introduces a novel short-term prediction model utilizing phase space reconstruction and HSSA-DNN, utilizing Bhattacharyya Distance for data cluster classification and enhanced moth-flame optimization for real-time applications.

4.4.2. Improved Moth-Flame Optimization Algorithm

The MFO algorithm employs a moth to conduct local searches, thereby balancing global exploration and local mining capabilities. Incorporating the linear inertia weight method improves upon the sine-cosine algorithm, thereby enhancing optimization potential and accelerating convergence, as depicted in Equation (22).

$$M_i = D_i \cdot e^{bt} \cdot cos(2\pi t) + w \cdot F_j \tag{22}$$

The MFO algorithm's convergence speed and global search capability are enhanced by a modified mechanism for updating moth positions, utilizing a hybrid search strategy and mutation operator.

The Bhattacharyya distance serves as a metric for assessing the similarity between two probability density functions, thereby enhancing the convergence rate. If we denote the two probability density functions as P and Q, then their Bhattacharyya distance is defined as Equation (23):

$$BD(P, Q) = -ln\big(BC(P, Q)\big) \tag{23}$$

The probability density function is derived using kernel density estimation. This study suggests substituting the probability density with variance, formulating the Bhattacharyya distance Equation (24) based on variance, and employing the Bhattacharyya distance to quantify the variance difference between the two probability distributions. $BD\big(D(X), D(Y)\big) = -ln\Big(BC\big(D(X), D(Y)\big)\Big)$ (24)

Where, $BC\big(D(X), D(Y)\big)$ The Bhattacharyya coefficient, for discrete probability distributions. However, during the iterative search process, when $d_i \le w \cdot D_E$, a linear search mechanism is introduced. The positions of moths are then updated as Equation (25):

$$M_i(l+1) = \begin{cases} F_i - A \cdot D_i', & i \le f_{no} \\ D_i \cdot e^{bt} \cdot cos(2\pi t) + F_{f_{no}}(l), & i > f_{no} \end{cases} \tag{25}$$

Where, D_i' and D_i are given as $D_i' = |C \cdot F_i - M_i|$ and $D_i = \left|F_{f_{no}} - M_i\right|$. Where W represents the weight coefficient, and its value is chosen as 0.1; $A = 2a \cdot R - a$; $C = 2 \cdot R$; $a = -1 + l * (-1/L)$; R A is a random constant within the range [0,1]. Adjusting the values of A and C enables reaching various locations around the flame relative to the current position.

Table 1: Pseudocode for Improved Moth Flame Optimization

Algorithm 1: Improved Moth Flame Optimization Algorithm

```
Randomly initialize each individual in moths using population in equation ()
$M_{ij} = lb_i + u_j(ub_i - lb_i)$
Initialize the iteration count $l = 1$;
while $l < L + 1$
Update $f_{no}$;
$OM = Fitness\ Function\ (M)$;
if $l == 1$
$F = sort(M)$; $OF = sort(OM)$;
else
$F = sort\big(M(l-1), M(l)\big)$; $OF = sort\big(OM(l-1), OM(l)\big)$;
end if
for $i = 1: n$
for $j = j: d$
Update $r$ and t; calculate D;
Update $M_{ij}$;
end for
end for
$l = 1 + 1$
end while
```

Based on this, the pseudo-code of the IMFO algorithm is shown in algorithm 1 and the steps of IMFO are illustrated in the above Table 1. However, the expression of w is shown in Equation (26).

$$w = (w_{max} - w_{min})^*(1 - (1/L)^2)^{\frac{1}{2}} \tag{26}$$

Where, the iteration times L and l stand for maximum and current, respectively. With each repetition, w changes nonlinearly from large to small. The iteration begins with random moth placement, allowing for global exploration. As information transmission increases, differences between individuals decrease. The low, declining w value allows for local exploration and mining, enhancing the month population's mining capabilities. The modified approach has been continuously tested, and it performs best when $w_{max} = 0.8$, $w_{min} = 0.3$.

5. Experimentation and Result Discussion

This research presents the evaluation findings for a proposed method using real-time datasets from Madurai and Chennai from January 1st to December 31st, 2021. The data includes hourly load values, temperature, humidity, wind speed, economic events, and public holidays. The model was evaluated in MATLAB to validate its effectiveness, minimizing cost functions based on parameter values and using the MATLAB R2022a programming language. The model predicts loads using weather, scheduling, and holiday information, historical loads, parallel PCANet and DWGRU components, and a sliding window. It divides datasets into training, validation, and testing sets, evaluating short-term load forecasting methods using MAPE and MAE (Equation (27)- (29)).

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i-\hat{y}_i}{y_i}\right| \times 100 \qquad (27)$$

$$MAE = \frac{1}{N}\sum_{i=1}^{N} y_i - \hat{y}_i \qquad (28)$$

$$RMSE = \sqrt{\frac{1}{T}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2}$$

(29)

Here, N represents the total number of testing samples, y_i denotes the actual load value for a specific hour, $\hat{y}_i$ represents the forecasted load value, and so forth.

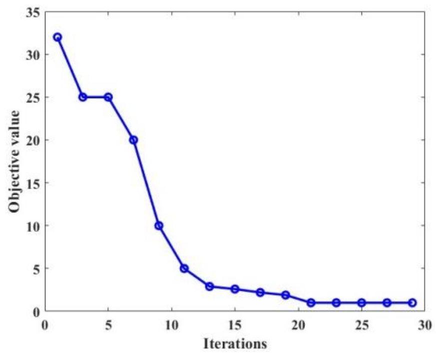


Fig. 3. Objective Value vs. Number of Iterations

Figure 3 shows IMFO algorithm results show higher connection leads to higher convergence rates, while far sparser topologies can still achieve acceptable long-term performance.

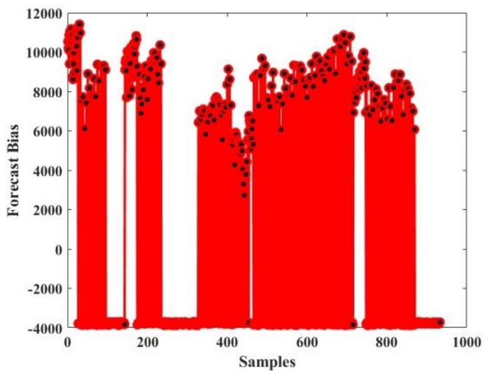


Fig. 4. Forecast Bias Graph

Figure 4 shows a forecast bias graph for 1000 samples, highlighting the importance of understanding load demand trends, including external factors like the economy, weather, and time index. Time index data is the easiest, but obtaining such factors is challenging.

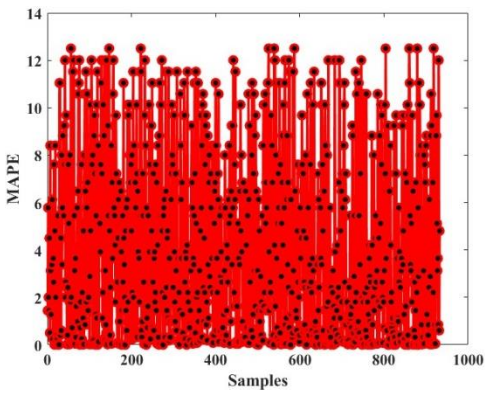


Fig. 5(a). MAPE Evaluation Results for Load Forecasting

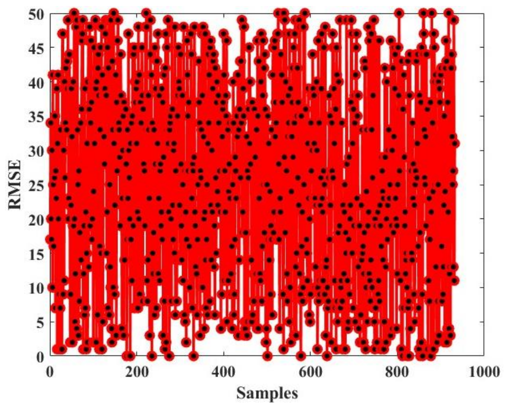


Fig. 5(b). RMSE Evaluation Results for Load Forecasting

Figures 5 displays MAPE and RMSE error distributions, plotted by DNN candidate model and historical pre-dispatch forecast. The lowest error is 0.092 for MAPE and 10 for RMSE, with outliers indicating significant errors.

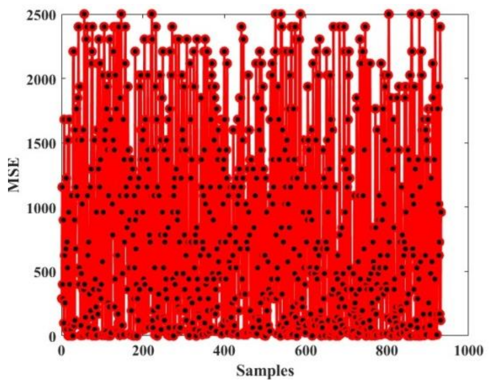


Fig. 6. Mean Squared Error Plot

Figure 6 above shows the MSE graph for the suggested work. The study uses real-time data from 20 samples over a year, showing train and test losses across iterations, indicating the model is not overfitting.

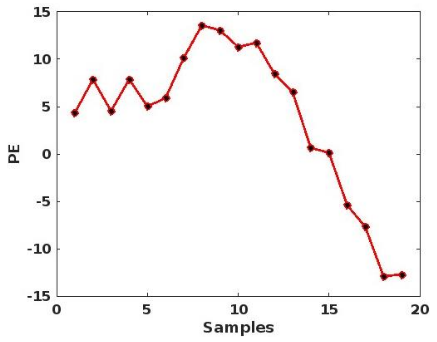


Fig. 7. Percentage Error Graph

Figure 7 depicts the anticipated percentage inaccuracy. More than 60% of errors are compounded between 5 and 10. The suggested distribution net load forecasting approach's accuracy and efficacy are shown by the forecast error analysis.

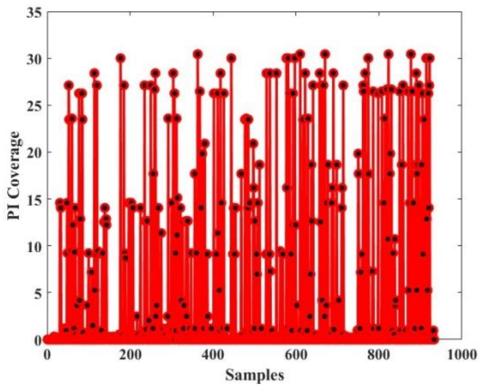


Fig. 8. PI Coverage Graph

Figure 8 shows a plot of PI coverage comparing the proposed model and benchmark models, indicating its stability and reliability, crucial for accurate decision-making based on accurate predictions.

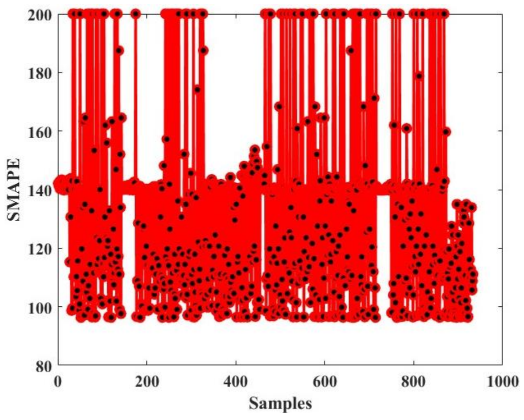


Fig. 9. SMAPE of Load Forecast Model

Figure 9 shows SMAPE for multiple models, considering 1000 real-time data samples. It collects temporal and spatial aspects from historical data and metadata, despite noise and mistakes.

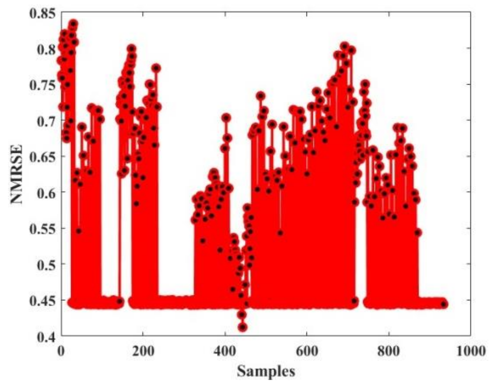


Fig. 10. Normalized RMSE Graph

The network's performance is not superior to random predictions, as shown in Figure 10. The NRMSE curves reveal significant variation in prediction performance based on the test sample.

Table 2: Comparison Results

Techniques	Accuracy (%)	R^2	MAPE	RMSE
ARIMA [23]	93	0.85	-	40.12
FE-SAMF-WNN [24]	98	0.944	0.234	-
LSTM-FA [25]	-	0.95	-	35.26
LF-NSNP [26]	-	-	0.2487	-
Proposed	99.15	0.96	0.2088	33

Table 2 present proposed method outperforms existing methods in accuracy and regression performance.

6. Research Conclusion

The manuscript discusses the importance of various factors in load forecasting, including seasonal data, wind speed, temperature, and historical load patterns. Pre-processing techniques like zero-mean normalization and feature selection through PCANet are employed to enhance data quality. Feature extraction is achieved using DWGRU to incorporate spatial, temporal, and semantic features. A hybrid deep learning model incorporating optimization algorithms is introduced for short-term load forecasting. In particular, the proposed model is a hybrid sampling and self-attention with deep neural network (HSSA-DNN) aimed at forecasting peak loads at 30-minute intervals. The proposed approach is implemented using Matlab Simulink and achieves high accuracy, with a MAPE of 0.2088 and RMSE of 19. Various inputs including load demand profile and weather information are considered to improve model generalization. Comparative analyses reveal the effectiveness of the proposed approach compared to other models such as ARIMA and LSTM-FA, achieving a prediction accuracy of 99.15%. The study also assesses parameters to enhance error reduction and computational efficiency, emphasizing the cost-effectiveness of the proposed hybrid model. In summary, the research suggests that the HSSA-DNN model offers accurate and efficient predictions of peak loads.

Declaration of Conflicting Interests

The authors stated that they have no conflicts of interest regarding the research, authorship, and publication of this article.

Funding

The authors stated that they have no conflicts of interest regarding the research, authorship, and publication of this article.

References

[1] Huang, N., Wang, S., Wang, R., Cai, G., Liu, Y., and Dai, Q., “Gated spatial-temporal graph neural network based short-term load forecasting for wide-area multiple buses”. International Journal of Electrical Power & Energy Systems, 145, pp.108651, (2023). https://doi.org/10.1016/j.ijepes.2022.108651

[2] Yang, Y., Wang, Z., Zhao, S., and Wu, J., “An integrated federated learning algorithm for short-term load forecasting”. Electric Power Systems Research, 214, pp.108830, (2023). https://doi.org/10.1016/j.epsr.2022.108830

[3] Hua, H., Liu, M., Li, Y., Deng, S., and Wang, Q., “An ensemble framework for short-term load forecasting based on parallel CNN and GRU with improved ResNet”. Electric Power Systems Research, 216, pp.109057, (2023). https://doi.org/10.1016/j.epsr.2022.109057

[4] Zhang, T., Zhang, X., Chau, T.K., Chow, Y., Fernando, T., and Iu, H.H.C., “Highly accurate peak and valley prediction short-term net load forecasting approach based on decomposition for power systems with high PV penetration”. Applied Energy, 333, pp.120641, (2023). https://doi.org/10.1016/j.apenergy.2023.120641

[5] Sekhar, C., and Dahiya, R., “Robust framework based on hybrid deep learning approach for short-term load forecasting of building electricity demand”. Energy, pp.126660, (2023). https://doi.org/10.1016/j.energy.2023.126660

[6] Ribeiro, M.H.D.M., da Silva, R.G., Ribeiro, G.T., Mariani, V.C. and dos Santos Coelho, L., “Cooperative ensemble learning model improves electric short-term load forecasting”. Chaos, Solitons & Fractals, 166, pp.112982, (2023). https://doi.org/10.1016/j.chaos.2022.112982

[7] Ran, P., Dong, K., Liu, X., and Wang, J., “Short-term load forecasting based on ceemdan and transformer”. Electric Power Systems Research, 214, pp.108885, (2023). https://doi.org/10.1016/j.epsr.2022.108885

[8] Li, S., Kong, X., Yue, L., Liu, C., Khan, M.A., Yang, Z., and Zhang, H., “Short-term electrical load forecasting using hybrid model of manta ray foraging optimization and support vector regression”. Journal of Cleaner Production, pp.135856, (2023). https://doi.org/10.1016/j.jclepro.2023.135856

[9] Shahare, K., Mitra, A., Naware, D., Keshri, R., and Suryawanshi, H.M., “Performance analysis and comparison of various techniques for short-term load forecasting”. Energy Reports, 9, pp.799-808, (2023). https://doi.org/10.1016/j.egyr.2022.11.086

[10] Srivastava, A.K., Pandey, A.S., Houran, M.A., Kumar, V., Kumar, D., Tripathi, S.M., Gangatharan, S. and Elavarasan, R.M., “A Day-Ahead Short-Term Load Forecasting Using M5P Machine Learning Algorithm along with Elitist Genetic Algorithm (EGA) and Random Forest-Based Hybrid Feature Selection”. Energies, 16(2), pp.867, (2023). https://doi.org/10.3390/en16020867

[11] Heidarpanah, M., Hooshyaripor, F., and Fazeli, M., “Daily electricity price forecasting using artificial intelligence models in the Iranian electricity market”. Energy, 263, pp.126011, (2023). https://doi.org/10.1016/j.energy.2022.126011

[12] Zambrano-Asanza, S., Morales, R.E., Montalvan, J.A., and Franco, J.F., “Integrating artificial neural networks and cellular automata model for spatial-temporal load forecasting”. International Journal of Electrical Power & Energy Systems, 148, pp.108906, (2023). https://doi.org/10.1016/j.ijepes.2022.108906

[13] Panagiotou, D.K., and Dounis, A.I., “Comparison of Hospital Building’s Energy Consumption Prediction

Using Artificial Neural Networks, ANFIS, and LSTM Network". Energies, 15(17), pp.6453, (2022). https://doi.org/10.3390/en15176453

[14] Hu, Y., Li, J., Hong, M., Ren, J., and Man, Y., "Industrial artificial intelligence-based energy management system: Integrated framework for electricity load forecasting and fault prediction". Energy, 244, pp.123195, (2022). https://doi.org/10.1016/j.energy.2022.123195

[15] Rao, S.N.V.B., Yellapragada, V.P.K., Padma, K., Pradeep, D.J., Reddy, C.P., Amir, M., and Refaat, S.S., "Day-ahead load demand forecasting in urban community cluster microgrids using machine learning methods". Energies, 15(17), pp.6124, (2022). https://doi.org/10.3390/en15176124

[16] Bashir, T., Haoyong, C., Tahir, M.F., and Liqiang, Z., "Short-term electricity load forecasting using a hybrid prophet-LSTM model optimized by BPNN". Energy Reports, 8, pp.1678-1686, (2022). https://doi.org/10.1016/j.egyr.2021.12.067

[17] Tang, X., Chen, H., Xiang, W., Yang, J., and Zou, M., "Short-term load forecasting using channel and temporal attention based temporal convolutional network". Electric Power Systems Research, 205, pp.107761, (2022). https://doi.org/10.1016/j.epsr.2021.107761

[18] Liu, R., Chen, T., Sun, G., Muyeen, S.M., Lin, S., and Mi, Y., "Short-term probabilistic building load forecasting based on feature-integrated artificial intelligent approach". Electric Power Systems Research, 206, pp.107802, (2022). https://doi.org/10.1016/j.epsr.2022.107802

[19] Ghenai, C., Al-Mufti, O.A.A., Al-Isawi, O.A.M., Amirah, L.H.L., and Merabet, A., "Short-term building electrical load forecasting using adaptive neuro-fuzzy inference system (ANFIS)". Journal of Building Engineering, 52, pp.104323, (2022). https://doi.org/10.1016/j.jobe.2022.104323

[20] Meng, Z., Xie, Y., and Sun, J., "Short-term load forecasting using neural attention model based on EMD". Electrical Engineering, pp.1-10, (2022). https://doi.org/10.1007/s00202-021-01420-4

[21] Matrenin, P., Safaraliev, M., Dmitriev, S., Kokin, S., Ghulomzoda, A., and Mitrofanov, S., "Medium-term load forecasting in isolated power systems based on ensemble machine learning models". Energy Reports, 8, pp.612-618, (2022). https://doi.org/10.1016/j.egyr.2021.11.175

[22] Alrasheedi, A., and Almalaq, A., "Hybrid Deep Learning Applied on Saudi Smart Grids for Short-Term Load Forecasting". Mathematics, 10(15), pp.2666, (2022). https://doi.org/10.3390/math10152666

[23] Sayed, H.A., William, A., and Said, A.M., "Smart Electricity Meter Load Prediction in Dubai Using MLR, ANN, RF, and ARIMA". Electronics, 12(2), pp.389, (2023). https://doi.org/10.3390/electronics12020389

[24] ZulfiqAr, M., Kamran, M., Rasheed, M.B., Alquthami, T., and Milyani, A.H., "A Short-Term Load Forecasting Model Based on Self-Adaptive Momentum Factor and Wavelet Neural Network in Smart Grid". IEEE Access, 10, pp.77587-77602, (2022). DOI: 10.1109/ACCESS.2022.3192433

[25] Bacanin, N., Stoean, C., Zivkovic, M., Rakic, M., Strulak-Wójcikiewicz, R., and Stoean, R., "On the benefits of using metaheuristics in the hyperparameter tuning of deep learning models for energy load forecasting". Energies, 16(3), pp.1434, (2023). https://doi.org/10.3390/en16031434

[26] Li, L., Guo, L., Wang, J., and Peng, H., "Short-Term Load Forecasting Based on Spiking Neural P Systems". Applied Sciences, 13(2), pp.792. (2023). https://doi.org/10.3390/app13020792