

Artificial Lemming Algorithm-Optimized CrossKAN for Multi-Energy Load Forecasting

Zihao Lv

Jinan University

Jianning Yan

Guilin University of Electronic Technology

Fuzhao Hou

Guilin University of Electronic Technology

Ye Li

Xi'an Jiaotong-Liverpool University

Peng Sun

sunp0421@jnu.edu.cn

Jinan University

Article

Keywords: multi-energy load forecasting, CrossKAN, Kolmogorov–Arnold network, Artificial Lemming Algorithm, feature selection, time-series forecasting

Posted Date: October 21st, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-10883713/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Zihao Lv (1, equal contribution), Jianning Yan (2, equal contribution), Fuzhao Hou (2), Ye Li (3) and Peng Sun (1, corresponding author)

1 Jinan University, Guangzhou 510600, China

2 Guilin University of Electronic Technology, Guilin 541004, China

3 Xi'an Jiaotong-Liverpool University, Suzhou 215412, China

† These authors contributed equally. * Correspondence: sunp0421@jnu.edu.cn

Abstract

Accurate multi-energy load forecasting requires models that capture nonlinear, cross-variable and time-varying dependencies without imposing excessive computational cost. Here we present CrossKAN, a lightweight forecasting model whose hyperparameters are optimized using the Artificial Lemming Algorithm (ALA). A triple-correlation analysis first combines the Maximal Information Coefficient, SHAP importance and sliding-window correlation to rank candidate inputs according to statistical dependence, predictive contribution and temporal stability. CrossKAN retains the cross-correlation and patch-embedding backbone of CrossLinear, but replaces its linear forecasting head with a Kolmogorov–Arnold Network that learns nonlinear spline mappings. Across electricity, heating and cooling forecasting tasks, CrossKAN reduced MAE, RMSE and MAPE by 13.2–17.6% relative to the strongest baseline for each metric on the primary dataset. On Building Data Genome 2, the model reduced six MAE and RMSE measures by 1.2–12.0%. These results suggest that concentrating nonlinear modelling in the forecasting head can improve accuracy while preserving a lightweight encoder, within the datasets and evaluation settings considered here.

Keywords: multi-energy load forecasting; CrossKAN; Kolmogorov–Arnold network; Artificial Lemming Algorithm; feature selection; time-series forecasting

Introduction

Reliable short-term forecasts of electricity, heating and cooling demand are central to the operation of integrated energy systems, particularly as renewable generation increases. These loads are jointly shaped by weather, operating conditions and user behaviour, producing nonlinear, time-varying and cross-variable dependencies [1]. Forecasting models must therefore identify informative signals from heterogeneous inputs while representing complex temporal structure at a computational cost compatible with scheduling, energy allocation and demand response [2,3].

Multi-energy load forecasting spans time horizons from minutes to years, although the precise boundaries vary across studies [4]. Ultra-short-term and short-term forecasts support intraday operation, demand response and maintenance, whereas medium- and long-term forecasts primarily inform planning and capacity allocation. This study addresses short-term multi-step forecasting of electricity, heating and cooling loads from historical load observations and exogenous variables.

Existing approaches can be grouped broadly into statistical, conventional machine-learning and deep-learning methods. Linear regression [5], autoregressive integrated moving average models [6], exponential smoothing [7] and Kalman filters [8] are computationally efficient and interpretable. However, their linearity or stationarity assumptions can limit their ability to represent coupled load dynamics. Support vector machines [9], artificial neural networks [10] and random forests [11] relax some of these assumptions. Their performance nevertheless remains sensitive to feature construction and hyperparameter selection, and they do not explicitly model long-range temporal dependencies.

Deep-learning models learn nonlinear temporal representations directly from load sequences. Recurrent neural networks and gated variants such as long short-term memory networks [12] and gated recurrent units [13] propagate historical information through hidden states and have been widely applied to short-term load forecasting. Their sequential computation, however, restricts parallelism, and learning long-range dependencies remains difficult for long input windows.

Transformer-based models address long-range dependence by connecting distant time points through self-attention. Recent applications to energy forecasting include a hybrid iTransformer with two-stage feature extraction [14], a multiscale model of variable-time dependence [15] and a Bayesian Transformer for probabilistic multi-energy forecasting [16]. This flexibility comes with computational and memory costs that typically increase with sequence length, making predictive gains inseparable from training and deployment requirements.

Lightweight forecasting architectures based on multilayer perceptrons or linear projections offer a complementary route. Models such as DLinear, TSMixer and TimeMixer extract temporal structure by decomposing, mixing or projecting information across the temporal and variable dimensions. Their relatively simple computation makes them attractive when both forecasting accuracy and deployment cost matter.

CrossLinear [17] follows this lightweight design by using cross-correlation embedding to represent interactions between endogenous loads and exogenous variables, and patch embedding to encode local variation over an extended temporal context. Its forecasting head, however, remains a fixed linear projection. Consequently, the encoded features can only be recombined linearly at the output, which may be insufficient for peak-valley transitions, threshold effects and nonlinear weather-driven responses. Performance also depends on key hyperparameters, including the embedding dimension, hidden-layer width and learning rate.

Input selection presents a related challenge. A single association measure cannot simultaneously characterize nonlinear statistical dependence, a variable's contribution to the fitted predictor and the temporal stability of that relationship. Previous studies have used SHAP-based feature selection [18] or the Maximal Information Coefficient (MIC) in high-dimensional, small-sample settings [19], but these approaches emphasize model attribution and statistical dependence, respectively. We therefore combine MIC, SHAP and sliding-window correlation in a unified ranking procedure. This design is supported by multivariate feature-selection studies that balance relevance, redundancy and complementary dependence, together with evidence that cross-variable correlations may evolve over time [22,23,24].

To address these limitations, we developed a TCA-CrossKAN-ALA forecasting framework that integrates multi-view feature screening, lightweight temporal encoding, nonlinear output mapping and automated hyperparameter search. The study makes four contributions:

- 1) CrossKAN retains the cross-correlation and patch-embedding modules of CrossLinear, but replaces the linear forecasting head with a KAN layer. Learnable spline functions thereby model the nonlinear mapping from encoded temporal features to future loads without introducing a self-attention backbone.
- 2) Triple-Correlation Analysis (TCA) combines MIC, SHAP importance and sliding-window correlation to rank candidate inputs according to statistical dependence, predictive contribution and temporal stability.
- 3) ALA jointly searches the learning rate, hidden-layer width and embedding dimension [21]. It operates outside the forecasting network and does not alter the CrossKAN architecture.
- 4) The framework is evaluated for electricity, heating and cooling forecasting on a primary integrated-energy dataset and Building Data Genome 2 (BDG2), using comparisons with multiple forecasting models and hyperparameter-optimization algorithms.

Methods

Triple correlation analysis

Multi-energy load forecasting requires both an effective representation of cross-variable and temporal dependencies and an input set that remains informative as operating conditions change. Any single association measure can overlook nonlinear dependence, model-specific predictive contribution or temporal instability. We therefore treat feature evaluation as a distinct stage before CrossKAN forecasting, reducing the influence of redundant exogenous variables on the subsequent lightweight encoder and nonlinear forecasting head.

Triple-Correlation Analysis (TCA) evaluates each candidate variable from three complementary perspectives: nonlinear statistical dependence, model-derived feature contribution and time-varying correlation. Normalized scores are combined to produce a single feature ranking. TCA selects the exogenous inputs, whereas CrossKAN encodes cross-variable temporal structure and generates the forecast; the two components therefore address distinct stages of the forecasting pipeline.

Nonlinear correlation analysis

Because multi-energy loads may depend nonlinearly on meteorological and operating variables, we first use the Maximal Information Coefficient (MIC) to quantify nonlinear dependence between each input feature and the target load.

MIC is grounded in mutual information theory and represents generalized dependence by searching for the optimal partition across multiple binning resolutions. For an input feature and target load, MIC is defined as follows: Eq. (1) gives this definition.

$$\text{MIC}(X_i, Y) = \max_{x,y} \frac{I(X_i; Y)}{\log \min(x, y)} \quad (1)$$

Among them, $I(X_i; Y)$ represents the mutual information between features and loads, and x and y are the numbers of rows and columns of the grid, respectively.

MIC ranges from 0 to 1, with larger values indicating stronger statistical dependence between a feature and the target load. Computing MIC separately for all candidate inputs therefore identifies variables with strong generalized dependence on the load.

Random-forest-based SHAP feature contribution analysis

Statistical dependence alone does not show how strongly a feature contributes to a fitted forecasting model. We therefore use SHapley Additive exPlanations (SHAP) with a random forest to quantify the model-derived contribution of each feature.

The random forest model can be expressed as a nonlinear mapping function: Eq. (2) gives the random-forest mapping.

$$\hat{Y} = f_{\text{RF}}(X) \quad (2)$$

The SHAP method is based on the Shapley value idea in game theory and decomposes the model prediction results into the sum of the marginal contributions of each input feature. For the i th feature, its SHAP value is defined as: Eq. (3) gives the feature-level SHAP value.

$$\phi_i = \sum_{S \subseteq F \setminus \{i\}} \frac{|S|!(|F|-|S|-1)!}{|F|!} [f(S \cup \{i\}) - f(S)] \quad (3)$$

Among them, F represents the entire feature set, and S is the subset that does not include feature i .

In actual calculations, this study uses the absolute mean of the feature SHAP values as the global importance index of the feature: Eq. (4) defines the global SHAP importance.

$$\text{SHAP}_i = \mathbb{E}(|\phi_i|) \quad (4)$$

This model-based measure quantifies each feature's global contribution to the fitted random forest and complements the statistical dependence measured by MIC.

Time-varying correlation analysis

- 5 Multi-energy load systems are non-stationary, and feature-load correlations may vary substantially over time. We therefore use a sliding-window analysis to represent the temporal evolution of these relationships.

Assume that the length of the sliding window is L ; in the t th time window, calculate the correlation coefficient between input feature X_i and target load Y : Eq. (5) gives the windowed correlation.

$$10 \quad \rho_i(t) = \text{corr}(X_i^{(t:t+L)}, Y^{(t:t+L)}) \quad (5)$$

where $\text{corr}(\cdot)$ represents the Pearson or Spearman correlation coefficient.

By gradually calculating through the sliding window over the entire time series, the evolution trajectory of the correlation between features and loads can be obtained. Furthermore, this study uses the statistical mean of the correlation coefficient $\rho_i(t)$ within the window as the time-varying correlation indicator: Eq. (6) defines the

15 time-varying correlation indicator.

$$\text{TVC}_i = \frac{1}{T} \sum_{t=1}^T |\rho_i(t)| \quad (6)$$

This indicator summarizes the temporal stability of the feature-load relationship across sliding windows.

TCA fusion score for input selection

- 20 To convert the complementary MIC, SHAP and time-varying correlation measures into a unified criterion for input selection, TCA normalizes and combines their scores. This aggregation preserves the interpretation of each component while producing a single ranking of candidate features.

First, normalize the MIC, SHAP importance and time-varying correlation indicators: Eq. (7) gives the normalized indicators.

$$\tilde{M}_i = \mathcal{N}(\text{MIC}_i), \tilde{S}_i = \mathcal{N}(\text{SHAP}_i), \tilde{T}_i = \mathcal{N}(\text{TVC}_i) \quad (7)$$

- 25 Among them, $\mathcal{N}(\cdot)$ represents the Min-Max normalization operation.

Subsequently, a triple correlation composite score is constructed: Eq. (8) defines the composite score.

$$\text{TCA}_i = \alpha \tilde{M}_i + \beta \tilde{S}_i + \gamma \tilde{T}_i \quad (8)$$

Among them, α , β and γ are the weight coefficients, which are 0.4, 0.4 and 0.2, respectively, satisfying: Eq. (9) gives the weight constraint.

$$30 \quad \alpha + \beta + \gamma = 1 \quad (9)$$

The resulting score combines nonlinear statistical dependence (MIC), model-derived predictive contribution (SHAP) and the temporal stability of correlation (TVC), thereby avoiding reliance on a single feature-selection criterion.

- 35 TCA determines the exogenous input set for CrossKAN. The selected variables enter the cross-correlation embedding together with the endogenous loads, after which the KAN-based forecasting head maps the encoded features nonlinearly to future loads. TCA and CrossKAN thus address input selection and forecasting, respectively, within a single pipeline.

CrossKAN temporal feature modelling framework

CrossKAN retains the lightweight cross-correlation embedding and patch-embedding backbone of CrossLinear while replacing its linear forecasting head with a KAN-based nonlinear forecasting head. Cross-correlation embedding represents interactions between endogenous loads and exogenous variables, whereas patch embedding compresses local temporal segments into a global sequence representation. The proposed modification concentrates nonlinear modelling capacity at the forecasting head while preserving an efficient encoder. The overall architecture is shown in Fig. 1.

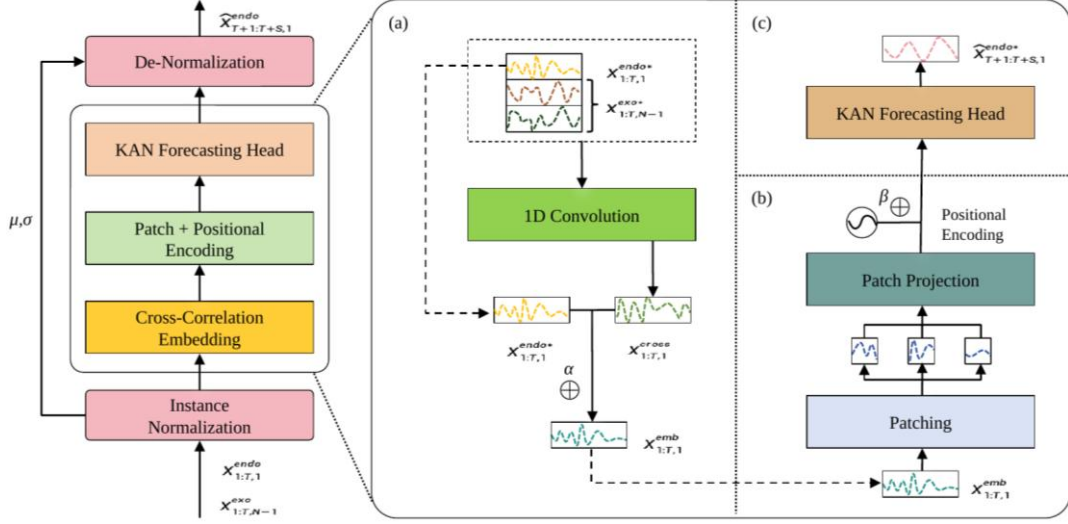


Fig. 1 | Overall architecture of CrossKAN.

Cross-correlation embedding layer

Multi-energy load forecasting uses endogenous electricity, heating and cooling loads together with exogenous meteorological and operating variables. Fully coupling all variables in a high-dimensional deep architecture may increase overfitting risk when samples are limited [20]. CrossKAN therefore uses a single one-dimensional convolution to represent endogenous-exogenous interactions with relatively few parameters and to provide a compact input for temporal modelling.

The model first stacks the normalized endogenous and exogenous variables. A single one-dimensional convolution then produces the cross-correlation representation $X_{1:T,1}^{cross}$. A learnable residual coefficient combines the original endogenous-load representation with this cross-correlation representation to obtain the embedding vector $X_{1:T,1}^{emb}$. The resulting representation retains changes in the target load and incorporates complementary information from exogenous factors. It therefore strengthens cross-variable feature extraction without substantially increasing model complexity. Eqs. (10) and (11) define the cross-correlation feature and its fusion with the endogenous embedding.

$$X_{1:T,1}^{cross} = \text{Conv1D}(\text{Stack}(X_{1:T,1}^{exo} - X_{1:T,1}^{emb})) \quad (10)$$

$$X_{1:T,1}^{emb} = \alpha \cdot X_{1:T,1}^{exo} + (1 - \alpha) \cdot X_{1:T,1}^{cross} \quad (11)$$

In the formula, Stack forms an $N \times T$ input matrix from the normalized endogenous and exogenous variables. Conv1D is a single-layer one-dimensional convolution that extracts direct relationships between these variables. The learnable fusion coefficient α ranges from 0 to 1. It adaptively balances the original endogenous representation and the cross-correlation representation in the final embedding.

Patch-embedding layer

The patch-embedding layer partitions the cross-correlation sequence into fixed-length local segments and projects each segment linearly. This operation compresses the sequence while retaining local fluctuations, allowing the forecasting head to combine short-term patterns across consecutive temporal segments. Patch construction is defined as follows: Eq. (12) gives the patch construction.

$$p_1^{endo}, p_2^{endo}, \dots, p_k^{endo} = \text{Patchify}(X_{1:T,1}^{emb}) \quad (12)$$

Here, P is the patch length, $k=T/P$ is the number of patches, and each segment $p_i^{endo} \in \mathbb{R}^{1 \times P}$.

Segmentation does not retain the absolute positions of patches in the original sequence. We therefore add positional encoding PE to the projected representation to preserve patch order. Fusing the patch representation with the positional encoding yields the final patch embedding: Eq. (13) defines the final patch embedding.

$$p^{endo} = \beta \cdot \text{Projection}_1(p_1^{endo}, \dots, p_k^{endo}) + (1 - \beta) \cdot PE \quad (13)$$

Here, Projection_1 maps each patch to a d -dimensional feature space, where d is the embedding dimension. $PE \in \mathbb{R}^{k \times d}$ denotes the positional-encoding matrix. The learnable coefficient β balances the data-driven patch representation against the positional encoding. This allows their relative contributions to adapt to the training data.

Linear forecasting head and its limitations

CrossLinear concatenates all patch embeddings in temporal order and applies a single linear projection to generate an S -step forecast. This short computational path and low parameter count contribute to the model's efficiency.

Note that the patch after splicing is expressed as $\text{Concat}(P^{endo})$, then the output of the original linear forecasting head $X_{T+1:T+S,1}^{endo*}$ can be expressed as: Eq. (14) gives the original linear-head forecast.

$$\hat{X}_{T+1:T+S,1}^{endo*} = \text{Projection}_2(\text{Concat}(P^{endo})) \quad (14)$$

The linear head efficiently combines encoded features, but its output remains a fixed linear combination. Its capacity may therefore be limited when load responses exhibit abrupt peak-valley transitions, threshold effects or complex coupling. CrossKAN addresses this limitation by replacing the linear head with a KAN-based nonlinear forecasting head while leaving the encoder unchanged.

KAN-based nonlinear forecasting head

After cross-correlation and patch embedding, the encoder produces features that represent cross-variable interactions and local temporal patterns. In CrossLinear, the concatenated representation is mapped to the output through a fixed linear transformation. Such a head cannot learn input-dependent nonlinear transformations, which may be important when coupled loads respond jointly to weather, operating conditions and abrupt local changes.

CrossKAN instead uses a KAN-based nonlinear forecasting head. KAN assigns learnable one-dimensional spline functions to connections, enabling data-driven local nonlinear transformations for different input dimensions before producing multi-step forecasts. This design concentrates nonlinear modelling capacity at the forecasting head while preserving the lightweight encoder.

Specifically, the splicing of the front-end output is denoted as z . This study keeps the cross-correlation embedding and patch embedding structures unchanged, and maps the original linear prediction: Eq. (15) gives the original linear mapping.

$$\hat{y} = Wz + b \quad (15)$$

Replaced by KAN-based nonlinear function mapping: Eq. (16) gives the KAN replacement.

$$\hat{y} = KAN(z) \quad (16)$$

Based on the Kolmogorov–Arnold representation idea, KAN represents the multidimensional input-output relationship as a combination of several learnable one-dimensional functions. For the KAN-based forecasting head used in this study, its calculation process can be written as: Eqs. (17) and (18) define the hidden and output mappings.

$$h_j = \sum_{i=1}^D \phi_{ij}(z_i), \quad j=1,2,\dots,H, \quad (17)$$

$$\hat{y}_k = \sum_{j=1}^H \psi_{jk}(h_j), \quad k=1,2,\dots,S, \quad (18)$$

In the formula, D is the input-representation dimension, H is the number of KAN hidden nodes, and S is the forecast horizon. The functions $\phi_{ij}(\cdot)$ and $\psi_{jk}(\cdot)$ are learnable one-dimensional splines. Unlike a linear forecasting head, this structure applies independent nonlinear transformations to different input dimensions. Function composition then produces multi-step forecasts. CrossKAN thus retains lightweight encoding of cross-variable and local temporal patterns while concentrating nonlinear modelling in the forecasting head. It does not require a self-attention backbone. The KAN-based forecasting head is illustrated in Fig. 2.

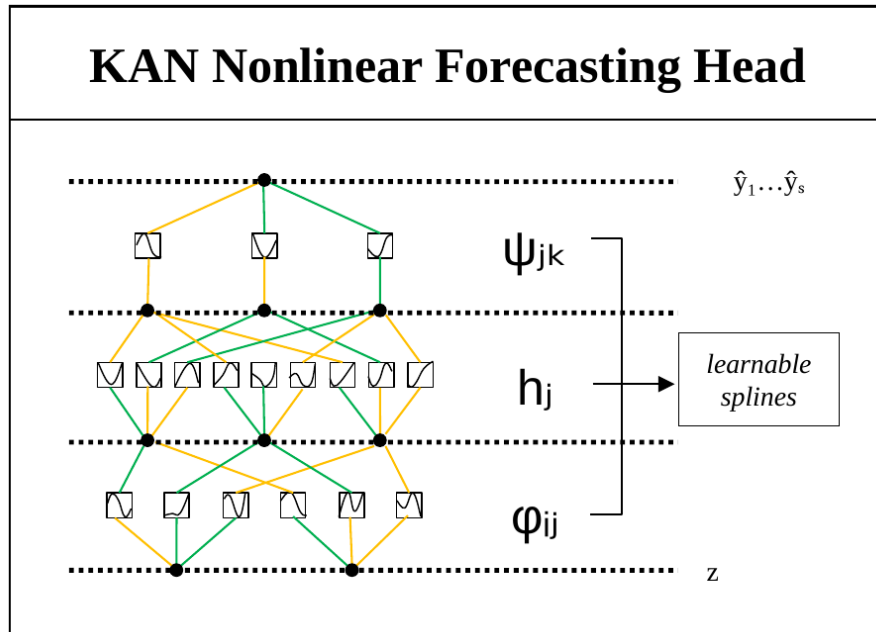


Fig. 2 | Architecture of the KAN-based forecasting head.

Artificial Lemming Algorithm

The Artificial Lemming Algorithm (ALA) is a population-based metaheuristic inspired by lemming migration, burrowing, foraging and escape behaviours [21]. In this study, ALA acts as an external hyperparameter optimizer that searches the learning rate, hidden-layer width and embedding dimension within predefined ranges. It does not modify the CrossKAN architecture.

Step 1: Population initialization

Assume that the lemming population consists of N lemming individuals, and the position of each lemming individual in the d -dimensional search space is expressed as Eq. (19).

$$Z_i = [z_{i1}, z_{i2}, \dots, z_{id}] \quad (19)$$

Then the initial lemming population can be expressed as: Eq. (20) defines the population matrix.

$$Z = \begin{bmatrix} z_{11} & z_{12} & \dots & z_{1d} \\ z_{21} & z_{22} & \dots & z_{2d} \\ \vdots & \vdots & \ddots & \vdots \\ z_{N1} & z_{N2} & \dots & z_{Nd} \end{bmatrix} \quad (20)$$

The positions of each dimension are randomly generated within the lower and upper bounds LB_j and UB_j , respectively: Eq. (21) gives the bounded initialization.

$$z_{ij} = LB_j + rand \times (UB_j - LB_j) \quad (21)$$

Among them, $rand$ is a random number in the interval $[0,1]$. This initialization method distributes the initial individuals within the given search boundary and provides different candidate starting points for subsequent searches.

10 Step 2: Long-distance migration strategy

Long-distance migration represents global exploration in ALA. It expands the search region during early iterations and reduces the risk of premature concentration around a local solution.

The position update of individual lemmings during the migration stage can be expressed as: Eq. (22) defines the migration update.

$$15 \quad Z_i(t+1) = Z_{best}(t) + F \cdot BM \cdot [R \cdot (Z_{best}(t) - Z_i(t)) + (1 - R) \cdot (Z_i(t) - Z_a(t))] \quad (22)$$

In the formula, the first position denotes the current optimal lemming, and the second denotes a randomly selected individual. BM is a Brownian-motion random vector that increases search diversity. The direction-adjustment factor and the random vector R , whose elements lie in $[-1,1]$, control the update direction. These perturbations expand the search range and reduce premature concentration around local optima.

20

Step 3: Burrowing strategy

Burrowing provides supplementary exploration around potentially promising regions of the search space.

Its position update formula is: Eq. (23) defines the burrowing update.

$$Z_i(t+1) = Z_i(t) + F \cdot L \cdot (Z_{best}(t) - Z_b(t)) \quad (23)$$

25 Among them, $Z_b(t)$ is a random lemming individual; $L = rand \times (1 + \sin(\frac{t}{2}))$ controls the burrowing intensity over iterations. This strategy guides the population to gradually concentrate in the advantageous area while maintaining search randomness.

Step 4: Foraging strategy

Foraging represents local exploitation around promising solutions and is modelled using a spiral search mechanism: Eq. (24) defines the foraging update.

30

$$Z_i(t+1) = Z_{best}(t) + F \cdot spiral - rand \cdot Z_i(t) \quad (24)$$

where the spiral factor is: Eqs. (25) and (26) define the spiral factor and radius.

$$spiral = r \cdot [\sin(2\pi \cdot rand) + \cos(2\pi \cdot rand)] \quad (25)$$

$$r = \sqrt{\sum_{j=1}^d (z_{best,j} - z_{i,j})^2} \quad (26)$$

Step 5: Predator avoidance and escape

5 When lemmings encounter the threat of predation, they will escape through fast movement and irregular paths. This behavior is used in the algorithm to enhance the ability to jump out of local extremes. Its position update formula is: Eq. (27) defines the escape update.

$$Z_i(t+1) = Z_{best}(t) + F \cdot G \cdot Levy(d) \cdot (Z_{best}(t) - Z_i(t)) \quad (27)$$

Here, $G = 2(1 - \frac{t}{T_{max}})$, and Levy flight simulates discontinuous large-step perturbations that help the algorithm escape local optima during later iterations.

10 Step 6: Energy decay and behavioural switching

To switch adaptively between exploration and exploitation, the lemming energy factor $E(t)$ is defined as: Eq. (28) defines the energy factor.

$$E(t) = 4 \cdot \arctan\left(1 - \frac{t}{T_{max}}\right) \cdot \ln\left(\frac{1}{rand}\right) \quad (28)$$

15 When $E(t) > 1$, the algorithm prioritizes migration and burrowing; when $E(t) \leq 1$, it performs foraging and escape. This mechanism emphasizes global exploration early in the search and gradually shifts towards local exploitation.

Step 7: Greedy selection

A forward greedy selection strategy compares fitness before and after each update and retains only the solution with better fitness for the next generation.

20 ALA-based hyperparameter optimization

ALA jointly searches the hidden-layer width, embedding dimension and learning rate of CrossKAN. These hyperparameters control the capacity of the KAN-based forecasting head, the dimensionality of the temporal representation and the training step size, respectively. ALA optimizes their configuration externally, whereas CrossKAN remains the forecasting model. The rationale for optimizing each parameter is as follows:

- 25 1) Hidden-layer width: This parameter controls the nonlinear capacity of the KAN-based forecasting head. Too few hidden units may underfit complex load dynamics, whereas too many can add redundant parameters and increase overfitting or training instability. Searching this width balances representation capacity and robustness.
- 30 2) Embedding dimension: This parameter determines the capacity and compression of the encoded temporal representation. An insufficient dimension may discard useful structure, whereas an excessive dimension may introduce redundancy. Its appropriate value can vary across datasets and load types.
- 35 3) Learning rate: This parameter governs training convergence. An excessively high value may cause oscillation or divergence, whereas an excessively low value slows convergence. Jointly searching the learning rate with architectural hyperparameters accounts for their interaction during training.

Figure 3 summarizes the ALA-based search procedure:

- 1) Parameter encoding and initialization. Encode the CrossKAN hidden-layer width, embedding dimension and learning rate as each lemming's position vector, and initialize the population within the specified bounds.

- 2) Fitness function. Use the forecasting mean squared error (MSE) on the training set to evaluate each candidate hyperparameter configuration.
- 3) Fitness evaluation. Train CrossKAN using each candidate configuration, compute its MSE and identify the best individual in the current population.
- 5 4) Migration and burrowing updates. During early iterations, update candidate positions using migration and burrowing to explore the hyperparameter space.
- 5) Boundary handling and discretization. Clip updated positions to the specified bounds, then round the hidden-layer width and embedding dimension to valid discrete values before fitness evaluation.
- 6) Foraging and escape updates. Apply local foraging search and Lévy-flight-based escape updates during the iterative search.
- 10 7) Greedy selection. Compare fitness before and after each update and retain the configuration with the lower MSE.
- ⑧ Judgement of iteration termination. We set the ALA population size to $N=10$, the problem dimension to $\text{dim}=3$ and the maximum number of iterations to $T_{\text{max}}=30$. The iteration limit is the stopping criterion. At iteration 30, the algorithm returns the best parameter combination found. Otherwise, it continues from the position-update stage.
- 15 9) Output. Use the best hyperparameter configuration found by ALA for final CrossKAN training and multi-energy load forecasting.

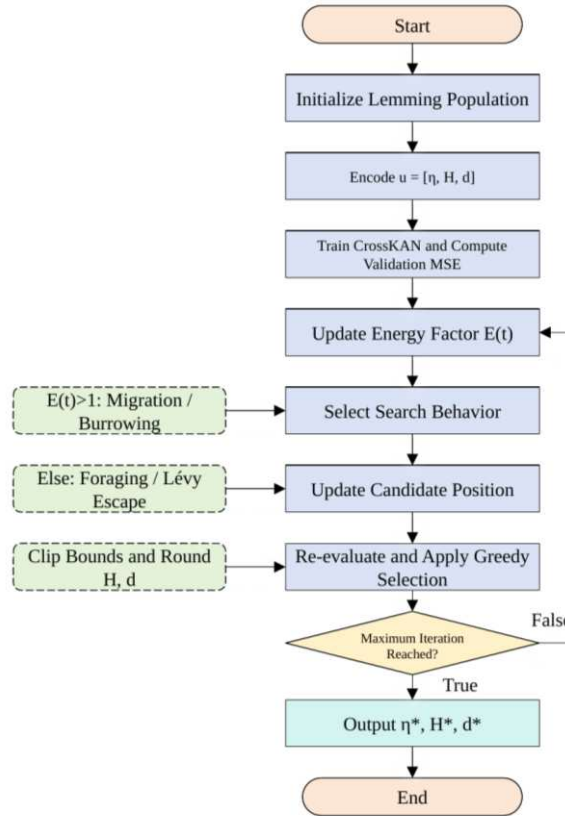


Fig. 3 | ALA-based hyperparameter optimization process.

ALA-optimized CrossKAN forecasting workflow

The complete framework comprises three distinct stages: TCA selects informative inputs, CrossKAN performs lightweight temporal encoding and nonlinear multi-step forecasting, and ALA searches key

hyperparameters. These modules operate sequentially as input analysis, core forecasting and parameter configuration (Fig. 4).

Step 1: Data preprocessing and feature construction. Impute missing values, normalize the data, and construct endogenous and exogenous sequences for each target. Convert the time series into supervised samples using a sliding window.

Step 2: TCA feature selection. Evaluate candidate exogenous variables using MIC, SHAP importance and sliding-window correlation, and use the selected variables as CrossKAN inputs.

Step 3: CrossKAN construction. Retain cross-correlation and patch embedding as the lightweight encoder, and replace the linear forecasting head with KAN spline functions for nonlinear multi-step forecasting.

Step 4: ALA hyperparameter optimization. Encode the hidden-layer width, embedding dimension and learning rate as candidate positions and minimize training-set MSE within the predefined ranges. ALA searches hyperparameters externally and does not alter the forecasting architecture defined in Step 3.

Step 5: Training and forecasting. Train CrossKAN using the configuration selected by ALA and generate S-step electricity, heating and cooling load forecasts for the test set.

Step 6: Evaluation. Evaluate forecasting errors using MAE, RMSE and MAPE, with R^2 additionally reported in the optimizer comparison. Comparisons with forecasting baselines and alternative optimizers assess TCA feature selection, the CrossKAN architecture and ALA search under the stated experimental settings.

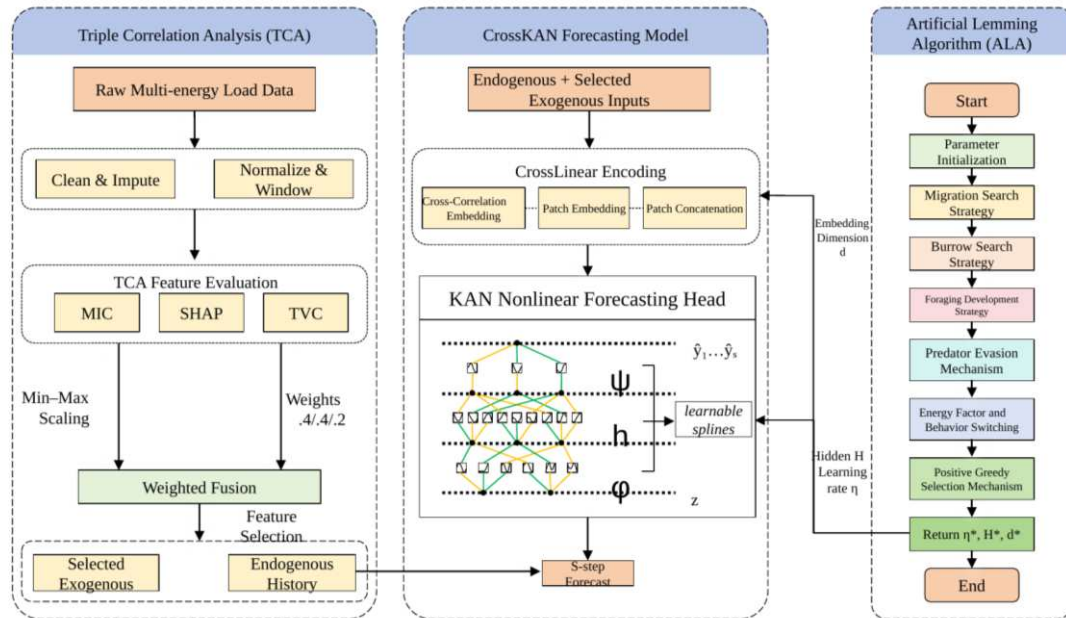


Fig. 4 | Workflow of ALA-optimized CrossKAN.

Results

Data description

We evaluated the proposed framework on two datasets from different sources. Both evaluations used the same division of functions among TCA feature selection, CrossKAN forecasting and ALA hyperparameter search. Normalization parameters were estimated from the training set only. Predictions were restored to the reporting scales defined below before MAE, RMSE and MAPE were calculated.

Primary dataset: Hourly integrated-energy-system data from Arizona State University's Tempe campus in 2023. The dataset contains 20 variables, including electricity, heating and cooling loads and indicators of residential, lighting, transportation and total energy use. To match the scales reported in the tables, electricity and cooling loads are divided by 10^2 , whereas heating load retains its original scale.

- 5 Cross-dataset evaluation: Building Data Genome 2 (BDG2), which covers 19 sites in North America and Europe from 2016 to 2017 and includes electricity, heating, chilled-water, steam, solar, water and irrigation meter data. We selected electricity, heating and cooling loads and their associated variables. Electricity retains its original scale, whereas heating and cooling are divided by 10 for reporting.

Experimental configuration

- 10 Experiments were conducted using an AMD R9-8945HX processor and an NVIDIA GeForce RTX 5070 Ti GPU. The implementation used Python 3.9 and PyTorch. The hyperparameter settings are listed in Table 1.

Table 1 | Hyperparameter settings

hyperparameters	value
prediction window	24
prediction step size	4
embedded dimensions	64
Hidden layer dimensions	100
Encoder-decoder layer	2
patch_len	16
Training test validation set ratio	7:2:1
Dropout inactivation rate	0.1
batch size	256
training rounds	50
Initial learning rate	0.0001
Early stopping strategy ratio	0.2
optimizer	Adam
Verification method	rolling window cross validation

Evaluation metrics

- 15 All experiments used fixed random seeds. Forecast accuracy was evaluated using mean absolute error (MAE), root mean squared error (RMSE) and mean absolute percentage error (MAPE); the coefficient of determination (R^2) was additionally reported for the optimizer comparison. MAE and RMSE use the load-specific reporting scales defined above, MAPE is expressed as a percentage, and R^2 is dimensionless.

Because reporting scales differ across loads and datasets, absolute MAE and RMSE values should not be compared across tasks. MAE, RMSE and MAPE are defined in Eqs. (29), (30) and (31), respectively.

$$MAE = \frac{1}{N} \sum_{i=1}^N |\hat{y}_i - y_i|$$

(11)

$MAE =$

5 $\frac{1}{N} \sum_{i=1}^N |\hat{y}_i - y_i|$

(29)

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

(12)

$RMSE =$

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

(30)

$$MAPE = \frac{100\%}{N} \sum_{i=1}^N \left| \frac{\hat{y}_i - y_i}{y_i} \right|$$

(13)

$MAPE =$

10

$$\frac{100\%}{N} \sum_{i=1}^N \left| \frac{\hat{y}_i - y_i}{y_i} \right|$$

(31)

In the formula, y_i is the actual value of test sample i ; $\hat{y}_i$ is the corresponding model prediction; and $\bar{y}$ is the mean of the observed values.

Correlation analysis and variable selection

15 We first applied TCA to the primary dataset because exogenous variables may contribute differently to electricity, heating and cooling forecasts. MIC, random-forest-based SHAP importance and sliding-window correlation were normalized and combined to rank candidate features and select CrossKAN inputs. This stage changes the input set but not the forecasting architecture.

Table 2 | TCA results for electrical load

Triple correlation analysis fusion heat map (feature importance, choose 6 out of 19)

	MIC	SHAP	TVC	Fusion
KWlightbulbs	0.98	0.99	0.99	0.986
KW#Houses	0.97	0.94	0.99	0.962
KWgalsgas	0.82	0.83	0.98	0.856
Combined Tons Carbon	0.65	0.89	0.76	0.768
Combined mmBTU	0.72	0.76	0.86	0.764
GHG	0.5	0.8	0.6	0.64

Table 3 | TCA results for heating load

Triple correlation analysis fusion heat map (feature importance, choose 6 out of 19)

	MIC	SHAP	TVC	Fusion
HTmmBTU#lightbulbs	0.98	0.99	0.99	0.986
HTmmBTU#Houses	0.97	0.79	1	0.904
HTmmBTUgalsgas	0.71	0.85	0.98	0.820
Totalgalsgas	0.67	0.85	0.88	0.784
Combined mmBTU	0.656	0.882	0.68	0.751
DOW	0.796	0.612	0.819	0.727

5 Table 4 | TCA results for cooling load

Triple correlation analysis fusion heat map (feature importance, choose 6 out of 19)

	MIC	SHAP	TVC	Fusion
CHWTON#lightbulbs	0.97	0.99	0.99	0.982
CHWTON#Houses	0.99	0.87	0.99	0.942
CHWTON#galsgas	0.91	0.69	0.99	0.838
Combined mmBTU	0.78	0.605	0.99	0.752
HTnmBTU galsgas	0.86	0.78	0.24	0.704
HTmmBTU	0.651	0.576	0.59	0.686

Tables 2-4 show that the household energy-use proxies (#Houses and #lightbulbs) received high TCA scores for all three loads. Within this dataset, these variables may represent dwelling-stock scale and underlying demand; their ranking should therefore be interpreted in the context of the current temporal resolution and fusion weights.

Aggregate energy-use indicators, including Combined mmBTU and carbon emissions, also received high TCA scores for heating and cooling. Their rankings may reflect co-variation between thermal demand and total energy use, but interpretation remains dependent on the variable definitions, temporal resolution and data source.

By contrast, variables such as GHG and DOW received relatively low TCA scores for the corresponding tasks. This ranking indicates limited relative contribution under the current data, forecast horizon and fusion weights; it does not exclude predictive value in other settings.

Effect of TCA feature selection

To isolate the effect of TCA, we compared CrossKAN using all candidate variables with the same model using only TCA-selected variables, while holding the architecture and training settings fixed.

Table 5 | Effect of TCA-based feature screening on forecasting errors

Electrical load				Heating load				Cooling load			
Input set	MAE	RMSE	MAPE	Input set	MAE	RMSE	MAPE	Input set	MAE	RMSE	MAPE
Contains all variables	4.167	6.28	2.967	Contains all variables	0.259	0.369	3.752	Contains all variables	1.263	1.73	3.729
Screened by TCA	3.524	5.145	2.513	Screened by TCA	0.256	0.355	3.605	Screened by TCA	1.182	1.623	3.388

As shown in Table 5, TCA reduced electricity-load MAE, RMSE and MAPE by 15.43%, 18.07% and 15.30%, respectively. For heating, the corresponding reductions were 1.16%, 3.79% and 3.92%; for cooling, they were 6.41%, 6.18% and 9.14%. The gain was largest for electricity and modest for heating, indicating task-dependent effects of feature selection.

Comparison of hyperparameter optimizers

We compared ALA with the Sparrow Search Algorithm (SSA), RIME, Whale Optimization Algorithm (WOA), Particle Swarm Optimization (PSO) and Crowned Porcupine Optimizer (CPO). Each method searched the same learning-rate, hidden-layer-width and embedding-dimension ranges ([10–4, 10–2], [32, 128] and [32, 128], respectively) using MSE as the fitness function. Table 7 lists the selected configurations, and Table 8 reports the resulting CrossKAN performance on electricity-load forecasting for the primary dataset. The common initial settings for all optimizers are listed in Table 6.

Table 6 | Initial settings of hyperparameters of all optimization algorithms

hyperparameters	value
N (population size)	10
Maxiter (maximum number of iterations)	30
dim (problem dimension)	3
lb (lower bound)	[0.0001, 32,32]
ub (upper bound)	[0.01, 128,128]
fitness function	MSE

Table 7 | Hyperparameters selected by each optimization algorithm

Optimization parameters	SSA	RIME	WOA	PSO	CPO	ALA
learning rate	0.0051	0.0036	0.0004	0.0065	0.0045	0.0014
Hidden layer	128	48	32	48	64	64
embedding layer	82	63	32	64	32	100

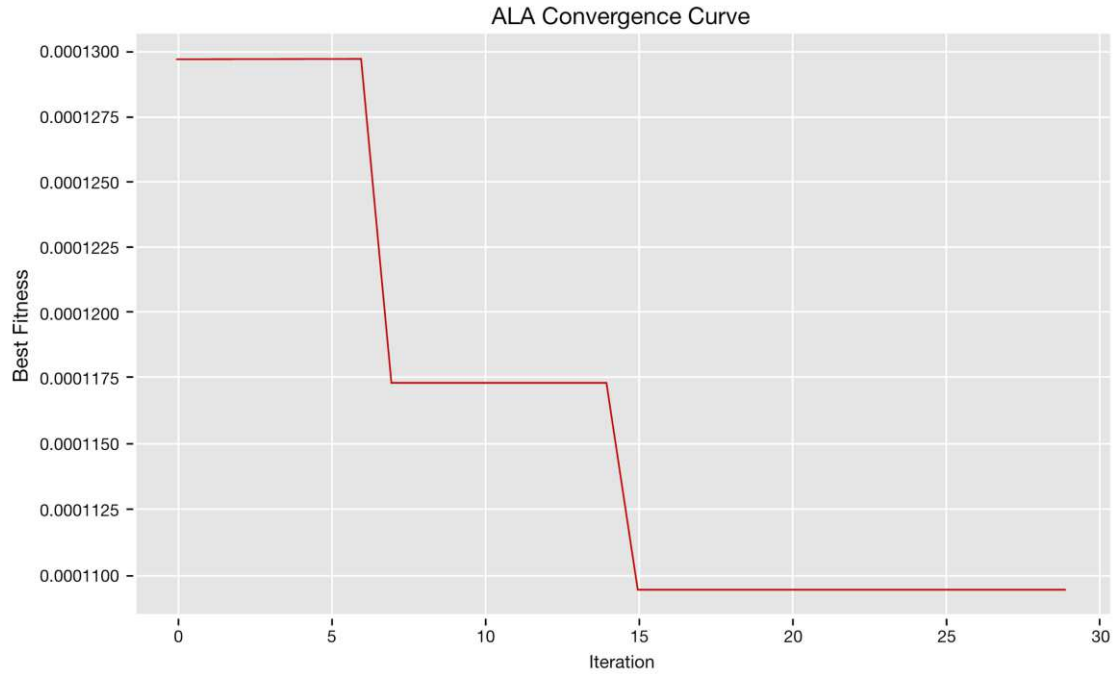


Fig. 5 | Convergence of ALA during hyperparameter optimization.

Figure 5 shows that the ALA fitness decreased sharply during the first few iterations, indicating rapid identification of configurations with lower objective values.

- 5 From iterations 5 to 10, the fitness decreased gradually and formed a plateau; the curve alone is insufficient to determine whether premature convergence occurred.

A further decrease occurred at approximately iteration 15, indicating that the search identified a candidate configuration with lower fitness.

- 10 After iteration 15, fitness remained stable, indicating convergence to a stable solution under the specified population size, iteration budget and random seed. This observation does not establish global optimality; Table 8 compares ALA with the alternative optimizers.

Table 8 | Performance of CrossKAN with hyperparameters selected by different optimization algorithms

Optimization algorithm	MAE	RMSE	MAPE	R ²
SSA	3.684	5.351	2.615%	0.952
RIME	3.589	5.223	2.527%	0.953
WOA	3.632	5.277	2.580%	0.953
PSO	3.799	5.511	2.709%	0.951
CPO	3.758	5.427	2.656%	0.951
ALA	3.524	5.145	2.513%	0.954

Forecasting results on the primary dataset

Forecast trajectories for electricity, heating and cooling loads are shown in Figs. 6–8.

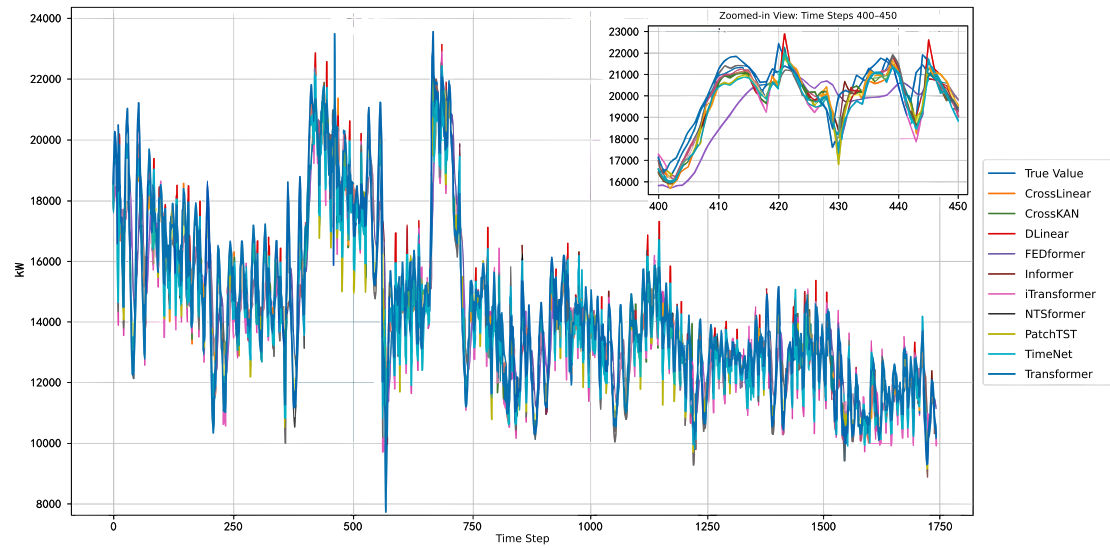


Fig. 6 | Forecasts of electricity load on the primary dataset.

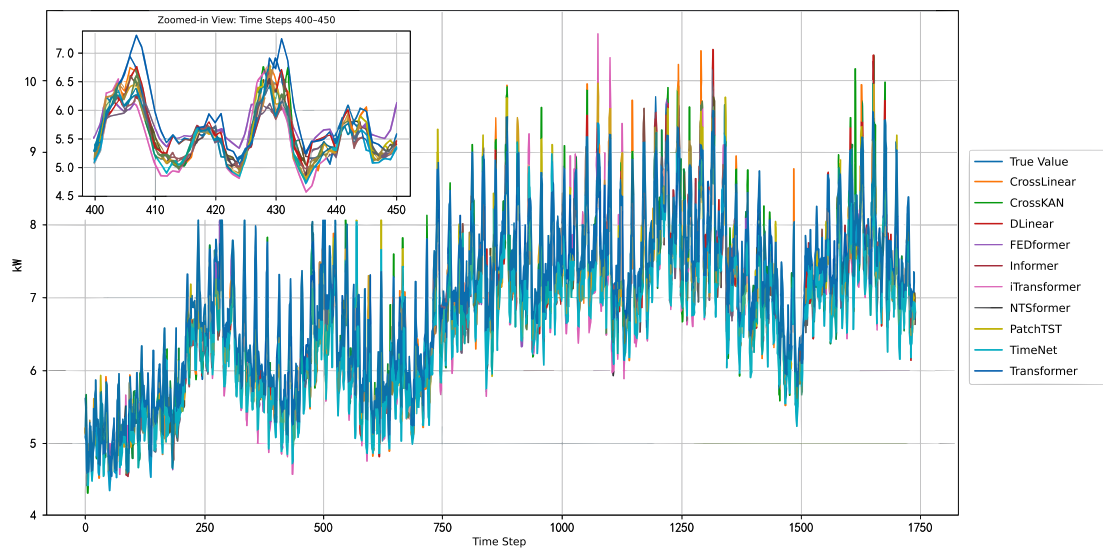


Fig. 7 | Forecasts of heating load on the primary dataset.

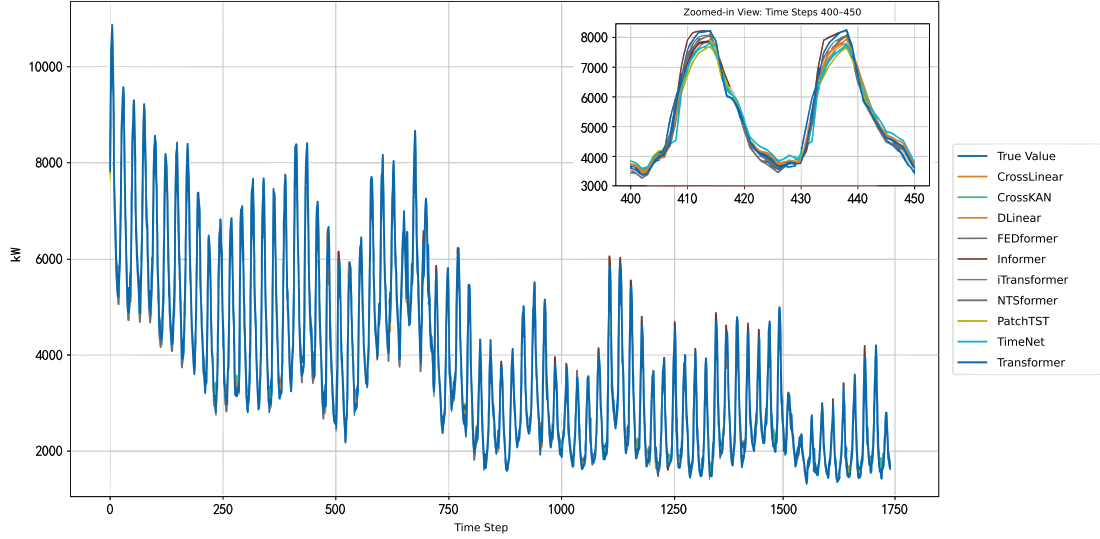


Fig. 8 | Forecasts of cooling load on the primary dataset.

CrossKAN achieved the lowest MAE, RMSE and MAPE for electricity, heating and cooling on the primary dataset. Relative to the strongest baseline for each metric, electricity-load MAE, RMSE and MAPE decreased from 4.141, 6.035 and 2.918% to 3.524, 5.145 and 2.513%, corresponding to reductions of 14.90%, 14.75% and 13.88%. Heating-load errors decreased from 0.301, 0.411 and 4.374% to 0.256, 0.355 and 3.605%, corresponding to 14.95%, 13.63% and 17.58%. Cooling-load errors decreased from 1.362, 1.894 and 3.921% to 1.182, 1.623 and 3.388%, corresponding to 13.22%, 14.31% and 13.59%. These values use the reporting scales defined above and agree with Table 9.

CrossKAN preserves CrossLinear's lightweight encoding of cross-variable and local temporal structure while adding nonlinear spline mapping at the forecasting head. Across the nine primary-dataset error measures in Table 9, its reductions relative to the strongest baseline ranged from 13.22% to 17.58%, consistent with the 13.2-17.6% range in the Abstract. This evidence supports the proposed design within the dataset, forecast horizon and metrics examined.

Table 9 | Performance of different baseline models on the primary dataset

Electrical load				Heating load				Cooling load			
Model	MAE	RMSE	MAPE	Model	MAE	RMSE	MAPE	Model	MAE	RMSE	MAPE
FEDformer	9.458	12.155	6.772%	iTransformer	0.379	0.513	5.435%	Informer	2.101	2.820	5.550%
Informer	6.485	8.938	4.576%	DLinear	0.376	0.495	5.437%	Transformer	1.867	2.553	4.791%
Transformer	5.530	7.180	3.922%	NTSformer	0.364	0.488	5.181%	TimeNet	1.866	2.617	5.580%
NTSformer	4.702	6.842	3.306%	PatchTST	0.361	0.490	5.211%	PatchTST	1.755	2.354	5.134%
PatchTST	4.520	6.469	3.192%	TimeNet	0.360	0.480	5.208%	DLinear	1.665	2.303	4.764%
TimeNet	4.362	6.386	3.040%	Transformer	0.360	0.439	5.428%	FEDformer	1.509	2.315	4.256%
DLinear	4.351	6.475	3.039%	Informer	0.335	0.445	4.906%	iTransformer	1.508	1.995	4.289%
iTransformer	4.303	6.416	3.054%	FEDformer	0.319	0.413	4.721%	NTSformer	1.448	1.936	3.993%
CrossLinear	4.141	6.035	2.918%	CrossLinear	0.301	0.411	4.374%	CrossLinear	1.362	1.894	3.921%
CrossKAN	3.524	5.145	2.513%	CrossKAN	0.256	0.355	3.605%	CrossKAN	1.182	1.623	3.388%

Cross-dataset evaluation on BDG2

We evaluated CrossKAN on BDG2 to examine performance under a different data source. BDG2 spans 19 sites in North America and Europe from 2016 to 2017 and differs from the primary dataset in load composition and the prevalence of zero or near-zero demand. This analysis assesses cross-dataset robustness under the stated settings rather than cross-region generalization.

BDG2 evaluation settings

The BDG2 evaluation used the same model structure, training strategy and hyperparameter-search procedure as the primary-dataset experiment. Because BDG2 contains many zero-load periods, particularly

for heating and cooling outside their operating seasons, MAPE is infinite for some load-model combinations and lacks a meaningful interpretation when the true value is zero. We therefore emphasize MAE and RMSE and report MAPE only as a qualitative reference.

BDG2 results

- 5 Forecast trajectories for electricity, heating and cooling loads on BDG2 are shown in Figs. 9–11.

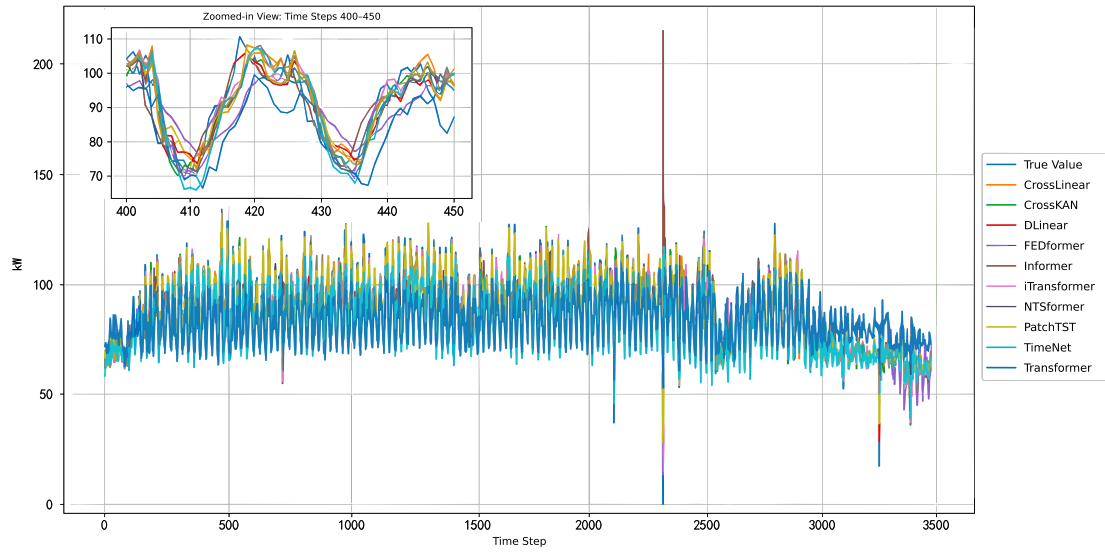


Fig. 9 | Forecasts of electricity load on the BDG2 dataset.

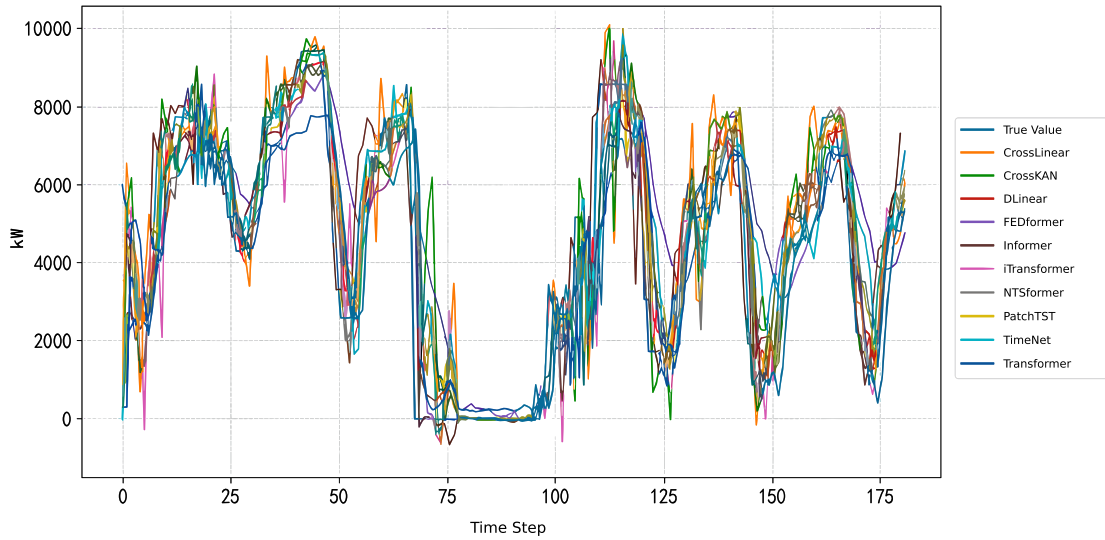


Fig. 10 | Forecasts of heating load on the BDG2 dataset (selected test-period segment shown for clarity).

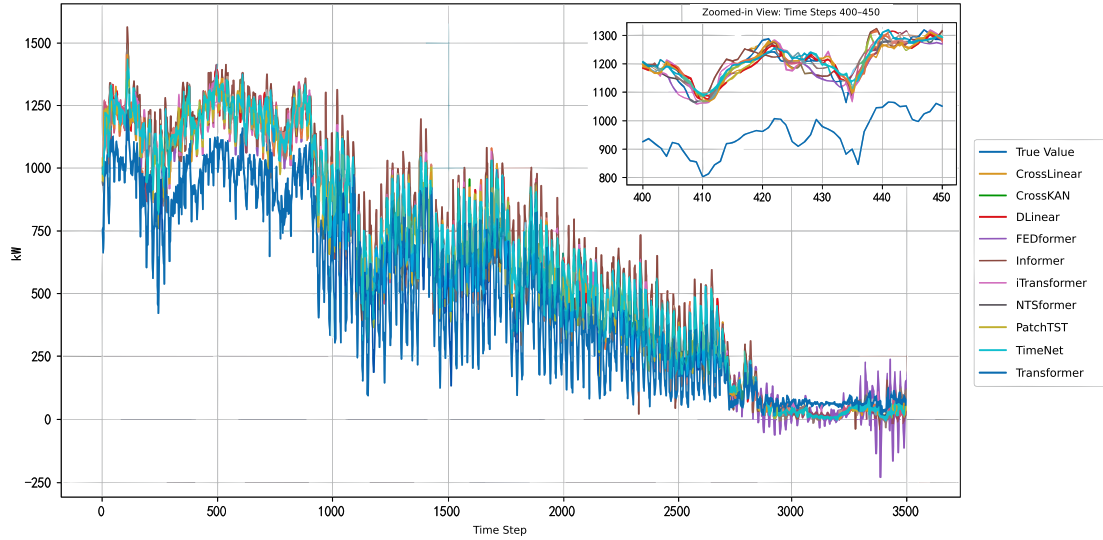


Fig. 11 | Forecasts of cooling load on the BDG2 dataset.

Errors were higher on BDG2 than on the primary dataset, although their different reporting scales preclude direct quantitative comparison of MAE and RMSE across datasets. The performance difference may reflect the following characteristics:

- 1) load patterns differ substantially across building sites, producing a non-stationary data distribution;
- 2) frequent zero-load intervals and abrupt transitions increase forecasting difficulty; and
- 3) unobserved site-specific factors may weaken periodic load patterns.

CrossKAN achieved the lowest reported MAE and RMSE for electricity, heating and cooling loads on BDG2 (Table 10). The magnitude of improvement varied across load types and baseline models, including DLinear, CrossLinear, Informer, FEDformer and PatchTST. These results indicate that the lightweight CrossLinear encoder combined with a nonlinear KAN forecasting head remained effective after changing the data source. Because the evaluation did not use leave-site-out or cross-region splits, however, the results do not establish generalization to unseen buildings or regions.

Table 10 | Performance of baseline models on the BDG2 dataset.

Electrical load			Heating load			Cooling load		
Model	MAE	RMSE	Model	MAE	RMSE	Model	MAE	RMSE
Transformer	8.903	10.71	FEDformer	45.108	64.81	Transformer	16.861	19.562
FEDformer	7.834	9.721	Transformer	26.562	40.524	FEDformer	5.015	6.388
TimeNet	5.496	7.443	NTSformer	10.911	39.542	Informer	4.762	6.560
DLinear	5.184	7.046	TimeNet	10.118	39.749	NTSformer	4.016	5.476
PatchTST	5.169	7.101	Informer	8.616	37.996	TimeNet	3.752	5.068
NTSformer	5.163	7.046	DLinear	8.086	38.292	DLinear	3.302	4.305
iTransformer	5.000	6.984	iTransformer	7.191	37.761	PatchTST	3.126	4.239
Informer	4.731	6.567	PatchTST	6.571	37.239	iTransformer	2.759	3.829
CrossLinear	4.724	6.585	CrossLinear	6.231	36.598	CrossLinear	2.685	3.718
CrossKAN	4.155	5.797	CrossKAN	6.046	35.399	CrossKAN	2.646	3.674

Discussion

This study introduced a TCA–CrossKAN–ALA framework to jointly address feature selection, lightweight temporal encoding, nonlinear output mapping and hyperparameter search in multi-energy load forecasting. TCA ranks exogenous variables using statistical dependence, model contribution and temporal stability.

CrossKAN retains the cross-correlation and patch-embedding backbone of CrossLinear while replacing the linear forecasting head with learnable KAN spline functions, and ALA searches the learning rate, hidden-

layer width and embedding dimension. Under the datasets and evaluation settings used here, CrossKAN achieved the lowest nine error measures on the primary dataset and the lowest six MAE and RMSE measures on BDG2. Relative to the strongest baseline for each metric, the corresponding reductions were 13.2–17.6% and 1.2–12.0%, respectively. The results support concentrating nonlinear modelling in the forecasting head while retaining a lightweight encoder.

Several limitations define the scope of these findings. First, the BDG2 evaluation changed the data source but did not use leave-site-out or cross-region splits, and therefore cannot establish broad cross-building generalization. Second, the analysis focused on point-forecast errors and did not quantify predictive uncertainty or performance under extreme-load conditions. Third, the TCA fusion weights and ALA search ranges were fixed, and their sensitivity to forecast horizon and operating regime remains unknown. Future work should prioritize independent-site validation, probabilistic forecasting and systematic sensitivity analyses of the feature-fusion and search settings.

Acknowledgements

This research was funded by the National Natural Science Foundation of China (22209058) and the Guangdong Basic and Applied Basic Research Fund (2023A1515011994).

Data availability

The datasets analysed in this study are publicly available. The primary integrated-energy dataset (ASU_Data) is available in the GitHub repository at https://github.com/hepei002/ASU_Data. The Building Data Genome Project 2 (BDG2) dataset used for the cross-dataset evaluation is available at <https://github.com/buds-lab/building-data-genome-project-2>.

Code availability

The code supporting the findings of this study is available from the corresponding author upon reasonable request.

Author contributions

J.Y. contributed to Conceptualization, Methodology, Visualization, and Writing – original draft by conceiving the study, formulating the overall research question, designing the CrossKAN architecture, preparing the figures, and contributing to the original draft. Z.L. contributed to Software, Investigation, and Validation by developing the software and experimental pipeline, implementing the proposed methods, conducting model training and comparative experiments, and validating the computational results. F.H. contributed to Methodology, Software, and Writing – original draft by adapting and integrating the Artificial Lemming Algorithm for CrossKAN hyperparameter optimization, refining the optimization module, and drafting the corresponding sections. Y.L. contributed to Data curation, Validation, Formal analysis, and Writing – review & editing by curating and documenting the datasets, independently verifying the evaluation metrics and comparative results, interpreting the findings from the perspectives of data-driven energy management and operational decision-making, and critically reviewing and revising the manuscript. P.S. contributed to Supervision and Writing – review & editing by supervising the study, contributing to the interpretation of the results, and reviewing and editing the manuscript. J.Y. and Z.L. contributed equally to this work. All authors read and approved the final manuscript.

Competing interests

The authors declare no competing interests.

Use of artificial intelligence tools

During manuscript preparation, a generative artificial intelligence tool was used to assist Chinese-to-English translation and language editing. The authors reviewed and revised the resulting text, and remain responsible for the scientific accuracy, references, equations and final content.

5 References

- [1] XU W, ZHANG Z, XU Z. A hybrid HVDC topology suitable for large-scale pure clean energy power base transmission[J]. *Electric Power*, 2023, 56(4): 17-27.
- [2] WANG S, et al. Multi-energy load forecasting for regional integrated energy systems considering temporal dynamic and coupling characteristics[J]. *Energy*, 2020, 195: 116964.
- 10 [3] KIM N, et al. Short-term electrical load forecasting with multidimensional feature extraction[J]. *IEEE Transactions on Smart Grid*, 2022, 13(4): 2999-3013.
- [4] ZHANG S, et al. Combing data-driven and model-driven methods for high proportion renewable energy distribution network reliability evaluation[J]. *International Journal of Electrical Power & Energy Systems*, 2023, 149: 108941.
- 15 [5] KAYTEZ F, et al. Forecasting electricity consumption: a comparison of regression analysis, neural networks and least squares support vector machines[J]. *International Journal of Electrical Power & Energy Systems*, 2015, 67: 431-438.
- [6] LEE C M, KO C N. Short-term load forecasting using lifting scheme and ARIMA models[J]. *Expert Systems with Applications*, 2011, 38(5): 5902-5911.
- 20 [7] HYNDMAN R J, et al. A state space framework for automatic forecasting using exponential smoothing methods[J]. *International Journal of Forecasting*, 2002, 18(3): 439-454.
- [8] TAKEDA H, TAMURA Y, SATO S. Using the ensemble Kalman filter for electricity load forecasting and analysis[J]. *Energy*, 2016, 104: 184-198.
- [9] TAN Z, et al. Combined electricity-heat-cooling-gas load forecasting model for integrated energy system based on multi-task learning and least square support vector machine[J]. *Journal of Cleaner Production*, 2020, 248: 119252.
- 25 [10] COCCIA G, et al. Artificial-neural-network-based model predictive control to exploit energy flexibility in multi-energy systems comprising district cooling[J]. *Energy*, 2021, 222: 119958.
- [11] LAHOUAR A, SLAMA J B H. Day-ahead load forecast using random forest and expert input selection[J]. *Energy Conversion and Management*, 2015, 103: 1040-1051.
- 30 [12] BATRA R, et al. Integration of LSTM networks with gradient boosting machines (GBM) for assessing heating and cooling load requirements in building energy efficiency[J]. *Energy Exploration & Exploitation*, 2024, 42(6): 2191-2217.
- [13] ZHANG Z, et al. Research on ultra-short-term load forecasting method of oil and gas field integrated energy system based on hybrid neural network[J]. *Electrical Engineering*, 2025, 107(10): 14021-14036.
- 35 [14] PEI J, et al. Tackling the duck curve in renewable power system: a multi-task learning model with iTransformer for net-load forecasting[J]. *Energy Conversion and Management*, 2025, 326: 119442.
- [15] DAI R, et al. VTformer: a novel multiscale linear transformer forecaster with variate-temporal dependency for multivariate time series[J]. *Complex & Intelligent Systems*, 2025, 11(6): 1-19.
- 40 [16] WANG C, et al. Probabilistic multi-energy load forecasting for integrated energy system based on Bayesian transformer network[J]. *IEEE Transactions on Smart Grid*, 2023, 15(2): 1495-1508.

- [17] ZHOU P, et al. CrossLinear: plug-and-play cross-correlation embedding for time series forecasting with exogenous variables[C]//Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining. V. 2. 2025.
- 5 [18] ZHENG Q, et al. TCN-GAT multivariate load forecasting model based on SHAP value selection strategy in integrated energy system[J]. Frontiers in Energy Research, 2023, 11: 1208502.
- [19] YU J, PAN R, ZHAO Y. High-dimensional, small-sample product quality prediction method based on MIC-stacking ensemble learning[J]. Applied Sciences, 2021, 12(1): 23.
- [20] BEJANI M M, GHATEE M. A systematic review on overfitting control in shallow and deep neural networks[J]. Artificial Intelligence Review, 2021, 54(8): 6391-6438.
- 10 [21] XIAO Y N, CUI H, KHURMA R A, et al. Artificial lemming algorithm: a novel bionic meta-heuristic technique for solving real-world engineering optimization problems[J]. Artificial Intelligence Review, 2025, 58(3): 84.
- [22] BROWN G, et al. Conditional likelihood maximisation: a unifying framework for information theoretic feature selection[J]. Journal of Machine Learning Research, 2012, 13: 27-66.
- 15 [23] SONG L, et al. Feature selection via dependence maximization[J]. Journal of Machine Learning Research, 2012, 13: 1393-1434.
- [24] ENGLE R. Dynamic conditional correlation: a simple class of multivariate generalized autoregressive conditional heteroskedasticity models[J]. Journal of Business & Economic Statistics, 2002, 20(3): 339-350.