Electrical Power and Energy Systems 175 (2026) 111575

Contents lists available at ScienceDirect

International Journal of Electrical Power and Energy Systems

journal homepage: www.elsevier.com/locate/ijepes

Transformer-based few-shot learning for modeling Electricity Consumption Profiles with minimal data across thousands of domains

Weijie Xia [a,*], Gao Peng [b], Chenguang Wang [c], Peter Palensky [a], Eric Pauwels [b], Pedro P. Vergara [a]

[a] Intelligent Electrical Power Grids (IEPG) Group, Delft University of Technology, Delft, 2628 CD, Netherlands
[b] Centrum Wiskunde & Informatica (CWI), Amsterdam, 9X42 JQ, Netherlands
[c] Alliander N.V, Arnhem, 6812 AH, Netherlands

ARTICLE INFO

Keywords:
Load profiles
Few-shot learning
Generative model
Mixture Density Network

ABSTRACT

Electricity Consumption Profiles (ECPs) are crucial for operating and planning power distribution systems, especially with the increasing number of low-carbon technologies such as solar panels and electric vehicles. Traditional ECP modeling methods typically assume the availability of sufficient ECP data. However, in practice, the accessibility of ECP data is limited due to privacy issues or the absence of metering devices. Few-shot learning (FSL) has emerged as a promising solution for ECP modeling in data-scarce scenarios. Nevertheless, standard FSL methods, such as those used for images, are unsuitable for ECP modeling because (1) these methods usually assume several source domains with sufficient data and several target domains. However, in the context of ECP modeling, there may be thousands of source domains, e.g., households with a moderate amount of data, and thousands of target domains, e.g., households that ECP are required to be modeled. (2) Standard FSL methods usually involve cumbersome knowledge transfer mechanisms, such as pre-training and fine-tuning. To address these limitations, this paper proposes a novel FSL framework that integrates Transformers with Gaussian Mixture Models (GMMs) for ECP modeling. The proposed approach is fine-tuning-free, computationally efficient, and robust even with extremely limited data. Results show that our method can accurately restore the complex ECP distribution with a minimal amount of ECP data (e.g., only 1.6% of the complete domain dataset) and outperforms state-of-the-art time series modeling methods in the context of ECP modeling.

1. Introduction

Electricity Consumption Profiles (ECPs) refer to the daily (or other specified periods) time series data of electricity usage, reflecting the volatility of human energy consumption behavior. ECP modeling involves understanding and modeling the complex distribution of ECP data. This modeling has significant applications in the energy sector. For instance, the modeled distribution of ECP for households or areas can be used to generate additional ECP data, aiding in electricity consumption prediction and load monitoring [1,2]. Understanding ECP distribution is also valuable for anomaly detection, risk analysis, energy supply, demand management, and energy system control [3–6]. With the prevalence of deep learning (DL), models such as generative adversarial networks (GANs), variational autoencoders (VAEs), diffusion, and flow-based models are adopted for ECP distribution modeling [7–10]. However, these modeling approaches are relatively 'data-hungry' and usually assume sufficient ECP training data exists in the target domains (e.g., households that ECP are required to be modeled).

In practice, access to ECP data in the target domain is often restricted due to several practical and regulatory constraints, including (1) metering infrastructure failures, such as malfunctions in smart meters or SCADA systems, which result in missing or corrupted time-series data [11], (2) privacy regulations, which limit the granularity and duration of data that Distribution System Operators (DSOs) are permitted to access—for instance, under Germany's digitalization framework, 15-minute resolution operational data is only retained for seven days to comply with privacy and data volume requirements [12,13], and (3) scarcity of original data, particularly in emerging applications or newly monitored regions, where historical ECP data is insufficient to train and validate data intensive models [14]. Therefore, an ECP modeling method that requires less data needs to be developed. In this paper, a domain refers to the ECP data collected from a terminal metering

* Corresponding author.
E-mail addresses: w.xia@tudelft.nl (W. Xia), P.P.VergaraBarrios@tudelft.nl (P.P. Vergara).

https://doi.org/10.1016/j.ijepes.2026.111575
Received 29 August 2025; Received in revised form 3 January 2026; Accepted 7 January 2026
Available online 19 January 2026

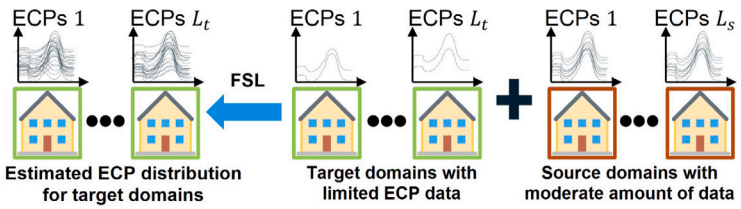


Fig. 1. Modeling residential ECP distribution across many domains (households) using FSL. L_s and L_t are the numbers of source domains (i.e., households with considerable data available) and target domains (i.e., households with limited data available), respectively.

device (e.g., in a residential household or building) that has a unique electricity consumption pattern due to human behavior differences.

Few-shot learning (FSL) has emerged as a promising solution for ECP modeling in data-scarce scenarios. As demonstrated in [15], even with a limited number of samples, it is possible to calibrate distributions effectively for classification tasks. FSL has been widely applied in images and audio generation [16,17]. This enlightens us to consider applying FSL in ECP modeling within data-scarce scenarios. Nevertheless, unlike a standard FSL task in image generation, in which there are usually several source domains with sufficient data and several target domains, ECP modeling often involves thousands of source domains (e.g., households) with moderate amounts of data and thousands of target domains (e.g., households). Furthermore, DL-based FSL typically requires model fine-tuning, which can be difficult to do effectively across thousands of domains in tasks like ECP modeling. These challenges underscore the need for an FSL method specifically designed for ECP modeling, as standard FSL methods may not suffice. Fig. 1 demonstrates the core idea of applying FSL for ECP modeling in this paper.

Gaussian Mixture Models (GMMs) are widely applied across various distribution modeling tasks, including ECP modeling [8]. The most common way to estimate the parameters of GMMs is the Expectation–maximization (EM) algorithm. The advantages of GMMs include (1) they are lighter in computational complexity compared to DL models, (2) it is a white-box model, and (3) similar to DL models, GMMs can theoretically approximate any distribution by increasing the number of components. Despite GMMs's advantages, GMMs as classical models seem isolated from FSL tasks. Moreover, some prior works have used DL model to predict mixture distribution parameters, which is called the Mixture Density Network (MDN), for flexible, conditional density estimation [18,19]. In this sense, combining the advantages of FSL, MDN, and GMMs can be seen as a promising idea. Recent work [20,21] has shown that it is possible to train a learner where one DL model is used to predict the parameters of another DL model. This inspires us to apply a DL model that can assist in the parameter estimation of GMMs with limited samples as inputs. Additionally, given the advantages of Transformers, such as their ability to capture dependencies and effectively knowledge transfer [22], we select the Transformer as our foundational DL architecture for GMM parameter estimation.

Inspired by the questions and insights described above, we propose a novel FSL method for ECP modeling. First, we propose a Transformer encoder architecture to acquire general knowledge from source domains (e.g., households with considerable data available). Then, we leverage this encoder to assist in the parameter estimation of GMMs in the target domains (e.g., households with limited data available). We interpret the knowledge learned from the source and target domains as shifts in the mean and variance of the Gaussian components in GMMs. To the best of our limited knowledge, this is the first research to propose the FSL method for ECP distribution modeling in data-scarce scenarios across thousands of source and target domains.[1] In summary, the contributions of this paper are

- From the ECP modeling perspective, we propose a novel method that requires significantly less data (e.g., less than 2% of the available dataset) to accurately model the complex ECP distribution. This method effectively addresses the challenges associated with ECP modeling in data-scarce scenarios.
- From the FSL perspective, we propose an efficient method for FSL tasks inspired by the MDN that combines a Transformer encoder architecture and GMMs. Compared with standard FSL, this method does not require fine-tuning in the target domain. To do this, we replace gradient descent-based models training/tuning with a significantly more efficient EM algorithm. As a result, the proposed method is suitable for applications in the energy sector and potentially extending to other Internet of Everything (IoE) applications that involve numerous source and target domains (terminal devices) [23].

2. Related work

2.1. Electricity consumption profile modeling

ECP modeling remains an active area of research in the energy domain, as accurate ECP models are vital for a wide range of applications, including load pattern calibration for distribution system reconfiguration [24–26], system operation optimization [27], system risk analysis [3], and other related tasks.

There are two primary approaches to modeling ECP datasets: the *bottom-up* and *top-down* approaches. The *bottom-up* approach builds consumption models from individual components [28], such as specific appliances (e.g., televisions, refrigerators). While this method can yield high-fidelity results, it requires detailed contextual information, such as the number of occupants, appliance types, and usage patterns, which is often difficult to obtain or generalize. In contrast, the *top-down* approach, which has gained greater popularity in both industry and research, leverages consumption data directly from smart meters (or similar metering devices) to develop data-driven models without considering the specific physical details. Most popular statistical or DL-based ECP modeling methods fall under the *top-down* category, as they rely solely on observed consumption patterns rather than detailed household-level features [29].

GMMs and Copula are widely used statistical methods for ECP modeling. In [30], GMMs are employed to model system load, demonstrating that despite their computational efficiency, GMMs can accurately capture the underlying load distribution. Similarly, in [31], the authors show that the proposed t-Copula framework effectively models the statistical properties of smart meter measurement datasets, highlighting its versatility in representing complex consumption patterns.

With the advancement of DL, an increasing number of DL-based methods have been proposed for ECP modeling. Recent works [8,32] have shown that the DL models excel at capturing the temporal correlations of ECP, which is crucial for planning the necessary future investment of flexible power distribution systems. With the prevalence of DL, many deep generative models have been applied in ECP modeling. For instance, [10] utilized a diffusion model for high-resolution, 1-minute level ECP modeling. Additionally, [7,33] employed conditional generative models to generate ECP data under varying weather

[1] The code and data of this project are available in Personal Repository and TU Delft Repository.

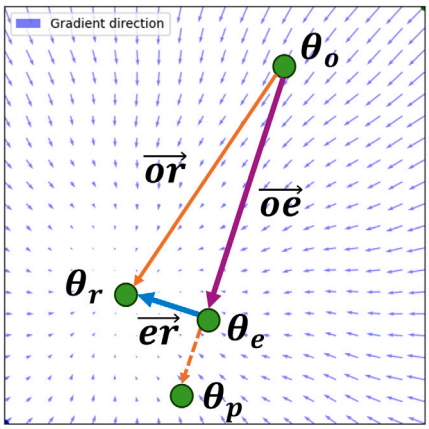


Fig. 2. Our proposed method begins with θ_o as the initial parameter of the GMMs. Let θ_r be the optimal parameter for the target domain (or the estimated parameter assuming a complete ECP dataset in the target domain). After applying the z-step EM algorithm on limited target-domain data, we obtain the estimated parameters θ_e. If the GMMs converge on this limited data, we achieve θ_p. Our method uses a Transformer to predict $\vec{er}$ such that $\theta_r = \vec{er} + \theta_e$.

conditions and customer characteristics. In [34], a GAN-based model is introduced to generate synthetic labeled load data, where labels refer to corresponding appliance usage patterns. The generated data closely resembled real-world labeled datasets and was effective for downstream model training. Furthermore, work in [35] proposes a hybrid VAE-GAN model to produce additional ECP data for smart home energy management systems, demonstrating improved data fidelity and utility.

However, we observe that current *top-down* ECP modeling methods typically assume sufficient target domain data is available (i.e., assuming sufficient training data for training a DL-based generative model). In practice, however, scenarios with limited source data are common in both industry and research, yet they are often overlooked in existing studies. In other research domains, data generation under limited data availability has recently become an active topic. For example, privacy-preserving data generation aims to improve data utility when only a small set of real samples is accessible, while ensuring that private information cannot be reconstructed [36], and a variety of generative approaches have been developed to address a similar challenge [37–39]. In the energy sector, particularly for residential customers, data are often sensitive and difficult to obtain due to privacy regulations and metering limitations. Within this context, ECP modeling in data-scarce scenarios remains an underexplored research problem in energy sector. The method proposed in this paper provides an effective solution for ECP modeling under data-scarce conditions, where current data-driven methods remain unexplored.

2.2. Few-shot learning

Standard FSL focuses on learning a discriminative classifier for tasks such as classification and detection [15,40–43]. For example, in [44], few-shot classification is performed by measuring the similarity between query samples and support samples through multi-branch semantic alignment of their spatial feature maps. The work in [45] introduces a semantic-guided augmentation technique that leverages pretrained generative models to produce class-preserving variant data, this approach yields improved end-task classification performance by injecting realistic, semantically relevant diversity into scarce datasets. Similarly, work in [46] propose an unsupervised few-shot representation learning framework that jointly enforces geometric invariance and pairwise consistency. By combining rotation-based self-supervision with contrastive learning, their method yields more discriminative and generalizable embeddings for downstream few-shot classification.

ECP modeling, which involves modeling the distribution of ECP data, is essentially an FSL generation task [23], and it has been extensively studied for data applications based on images, audio, and text [16,21,47,48]. A typical FSL method usually involves pre-training in the source domain and fine-tuning in the target domain [49]. This procedure can easily lead to overfitting in the target domain during fine-tuning. To address this, [50] proposed an elastic weight consolidation in the loss function to prevent overfitting. Similarly, [17] introduced a cross-domain correspondence mechanism to improve the diversity of model outputs and reduce overfitting. The work in [51] proposed a method to adaptively preserve the knowledge learned in the source domain, considering the target domain.

In addition to mitigating overfitting, a key challenge in FSL is how to effectively extract meaningful information from a limited number of target-domain samples. To address this, work in [52] proposed the Few-Gen framework, which emphasizes label-discriminative representations during the tuning process. This approach enables more efficient learning from limited samples when fine-tuning pretrained language models (PLM). Beyond improving learning efficiency, another complementary perspective is to enrich the information content available to the model. The work in [53] introduced a retrieval-based FSL framework, where similar samples from an external database are retrieved to augment the target domain data. This augmentation provides additional context and diversity, thereby enhancing the effectiveness of the fine-tuning process of PLM. Moreover, in [54], the author demonstrates that applying test-time training – where model parameters are temporarily updated during inference using a loss function derived from in-context examples – can significantly enhance the reasoning accuracy of PLM, achieving improvements of up to 6× in certain cases.

Despite the prevalence of FSL data applications based on images, audio, and other fields, its application to ECP modeling remains unexplored. Another significant challenge lies in the cumbersome nature of pre-training and fine-tuning for the vast number of domains anticipated in the energy sector. For instance, fine-tuning models individually for thousands of households is impractical. Therefore, a more scalable FSL method across domains for ECP modeling is required.

2.3. Mixture density network

The MDN framework uses a DL model to predict the parameters (mixing weights, component means, and covariances) of a parameterized mixture distribution conditioned on inputs [18,55–57], and MDNs for GMMs are typically trained by maximizing the conditional log-likelihood [58]. Several recent works address practical MDN limitations, for example, the work in [19] proposes a two-stage sampling-and-fitting pipeline in which a sampling network generates diverse hypotheses and a lightweight fitting network converts them into a stable parametric mixture, mitigating model collapse and improving likelihoods. The work in [59] shows that careful kernel design (a beta kernel for normalized wind power) enforces bounded support and reduces boundary leakage. And the work in [60] extends MDNs to graph-structured inputs by combining graph encoders with mixture-output heads, yielding better likelihoods on stochastic epidemic simulations and real-world regression tasks.

Despite these advances, integrating MDNs with FSL remains relatively underexplored. In this paper, we bridge that gap by proposing an MDN-inspired, Transformer-based estimator for GMM parameters that is tailored to few-shot ECP modeling.

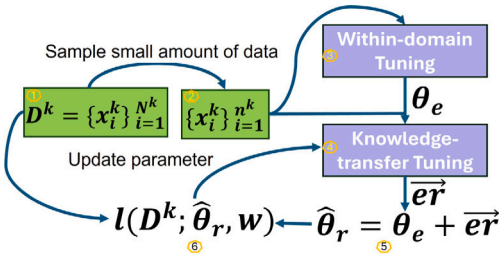


Fig. 3. Training/inference process of one domain. $\hat{\theta}_r = \{\mu_j, \Sigma_j\}_{j=1}^{J}$ represents the predicted parameters of GMMs, $\mathbf{w}$ is the weights of components, $l(\cdot)$ is the loss function. In the training process, $\mathcal{D}^k \in \mathcal{S}$. In the inference process, $\mathcal{D}^k \in \mathcal{T}$, and only $\hat{\theta}_r$ is predicted without loss computation and parameter updating for the Transformer. The index represents the order of the flowchart.

3. Problem formulation

3.1. Preliminaries

In ECP modeling, a typical daily ECP sample consists of T discrete time steps. For example, ECP data with a resolution of 60 min is characterized by a $T = 24$ time step (one for each hour), making one ECP sample a 24-dimensional point. Each dimension corresponds to a specific value of active power consumption of a time step. In general, the ECP dataset of kth domain (household) can be described as a

$$D^k = \{\mathbf{x}_i^k\}_{i=1}^{N^k} = \{(x_{1,i}^k, \dots, x_{T,i}^k)\}_{i=1}^{N^k}, \quad \mathbf{x}_i^k \in \mathbb{R}^{1\times T}, \tag{1}$$

where $x_{t,i}^k$ is the active power consumption of ith ECP sample (day) and kth domain at tth time step, $\mathbf{x}_i^k = (x_{1,i}^k, \dots, x_{T,i}^k)$ is the ith sample in kth domain, N^k is the number of samples (days) of kth domain. Therefore, the ECP dataset from all domains can be expressed as

$$\mathcal{D} = \{D^k\}_{k=1}^{K}, \tag{2}$$

where K is the number of domains (e.g., households).

3.2. Few-shot learning problem formulation

Assuming we have source domain collection $\mathcal{S} \subset \mathcal{D}$ and target domain collection $\mathcal{T} \subset \mathcal{D}$, where $\mathcal{S} \cap \mathcal{T} = \emptyset$. We aim to train a Transformer $f_{\theta 1}$ on $\mathcal{S}$, where we have access to a moderate amount of ECP data, denoted as N^{k_s}, for any k_s-th domain in $\mathcal{S}$. Our goal is to generalize $f_{\theta 1}$ to $\mathcal{T}$ to predict the parameters of GMMs with limited access to the data. For example, we sample n^{k_t} (n^{k_t} is a number) ECP samples from k_t-th domain in $\mathcal{T}$ as input of $f_{\theta 1}$. We aim for the GMMs with predicted parameters to represent the k_t-th domain's real ECP distribution accurately.

4. Proposed FSL method

Standard FSL methods usually require two steps (1) pre-training, which involves acquiring general knowledge, and (2) fine-tuning, which involves acquiring domain-specific knowledge. Our proposed method has similar processes but in reverse order. Fig. 2 shows the overall idea of our method. Let θ_o be the initial parameters of the GMMs, and θ_r be the optimal parameters of GMMs for a target domain (or the estimated parameters of GMMs assuming having complete ECP dataset of this domain). Our objective is to find the vector $\vec{or}$ in the parameter space such that

$$\theta_r = \vec{or} + \theta_o. \tag{3}$$

We write θ instead of $\vec{\theta}$ for simplicity. However, directly using a Transformer to predict $\vec{or}$ can be difficult and unstable, as the distance $\|\theta_o - \theta_r\|$ in the parameter space of GMMs can be very large. To address this issue, we propose an alternative method.

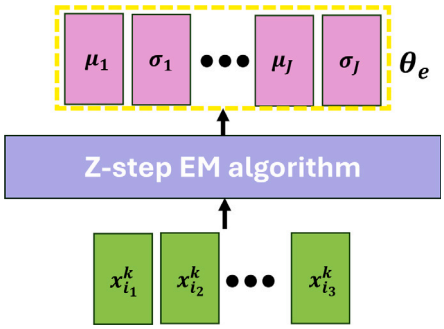


Fig. 4. z-step EM algorithm (Within-domain Tuning) aims at learning target-domain specific knowledge. x_i^k is the input ECP samples of kth domain and ith sample, in this process. J is the number of components of GMMs, each μ and Σ are the parameters of a Gaussian component in GMMs.

Algorithm 1 Within-domain Tuning of One Domain.

Require: Sampled ECP data $\{\mathbf{x}_i^k\}_{i=1}^{n^k} \subset D^k$, initial parameters θ_o, fixed weights $\mathbf{w}$, iteration steps z.

Initialize $\theta = \theta_o$
for $i = 1$ to z **do**
  E-step:
  for $j = 1$ to J **do**
    Compute the responsibility γ_{ij}:

$$\gamma_{ij} = \frac{w_j \, \mathcal{N}(\mathbf{x}_i^k \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}{\sum_{l=1}^{J} w_l \, \mathcal{N}(\mathbf{x}_i^k \mid \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l)}$$

  end for
  M-step:
  for $j = 1$ to J **do**
    Update the mean $\boldsymbol{\mu}_j$:

$$\boldsymbol{\mu}_j = \frac{\sum_{i=1}^{n^k} \gamma_{ij} \, \mathbf{x}_i^k}{\sum_{i=1}^{n^k} \gamma_{ij}}$$

    Update the covariance $\boldsymbol{\Sigma}_j$:

$$\boldsymbol{\Sigma}_j = \frac{\sum_{i=1}^{n^k} \gamma_{ij} (\mathbf{x}_i^k - \boldsymbol{\mu}_j)(\mathbf{x}_i^k - \boldsymbol{\mu}_j)^\top}{\sum_{i=1}^{n^k} \gamma_{ij}}$$

  end for
end for

First, we perform the z-step EM algorithm on the target domain data, which has a limited amount of data. Let θ_e be the estimated parameters of the GMMs after z-step EM as shown in Fig. 2. In this process, we do not aim for the GMMs to converge on the target domain data (reaching θ_p), but instead, stop early to achieve an approximately minimal $\|\vec{er}\|$. This step can be considered as fine-tuning in the target domain, with early stopping to prevent overfitting. Second, we train a Transformer encoder on the source domain to learn to predict $\vec{er}$. We can then compute $\vec{or}$ by

$$\vec{or} = \vec{oe} + \vec{er}. \tag{4}$$

In this process, $\vec{er}$ represents the transferred knowledge from the source domain, considering θ_e, and $\vec{oe}$ captures the target-domain specific knowledge. For convenience, we refer to the first process (z-step EM algorithm) as *Within-domain Tuning* and the second process as *Knowledge-transfer Tuning*.

Another intuition behind the design of Expression (4) is that constructing $\vec{or}$ through $\vec{or} = \vec{oe} + \vec{er}$ is easier than directly predicting $\vec{or}$

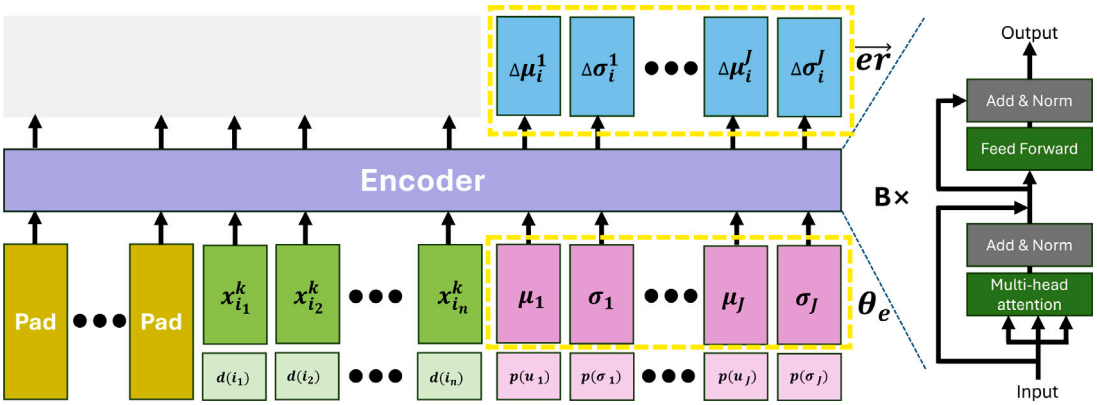


Fig. 5. Knowledge-transfer Tuning process. Sampled ECP samples are fed into the encoder together with corresponding θ_e to predict $\vec{er}$. $d(\cdot)$ represents the date embedding, which indicates the day of the year for x_i, and $p(\cdot)$ represents the parameter information, indicating which Gaussian component of the GMMs μ and σ belong to. We use the **[Pad]** token to align the shapes of inputs with different amounts n^k.

with a Transformer. Direct prediction implicitly requires the Transformer to approximate the entire EM process, which is a complex iterative procedure. In contrast, by decomposing $\vec{or}$ into a residual term, the Transformer only needs to model the part of the EM process from $\vec{oe}$ to $\vec{or}$, resulting in a simpler learning task. The ablation results in Section 6.1 further show how this design leads to improved performance.

Fig. 3 summarizes how Within-domain Tuning and Knowledge-transfer Tuning function during both training and inference. During training, we randomly sample a batch of source domains from $\mathcal{S}$. Next, from these source domains, we randomly sample a small number of ECP samples from each domain. These ECP samples are used in Within-domain Tuning to obtain the corresponding θ_e for each domain. Subsequently, the ECP samples and θ_e are fed into the encoder to predict the $\vec{er}$. The encoder is updated based on the loss described in Section 4.3. During inference, we follow the same procedure but use domain data from $\mathcal{T}$ to predict $\hat{\theta}_r$ without loss computation and parameter updating for the Transformer.

In the following sections, we provide a detailed explanation of the Within-domain Tuning and Knowledge-transfer Tuning processes.

4.1. Within-domain tuning

This section demonstrates how θ_e is obtained by Within-domain Tuning. As mentioned before, Within-domain Tuning is essentially the z-step EM algorithm of GMM on a small number of ECP samples.

In our method, we set the weights $\mathbf{w}$ of GMMs components to be fixed during the EM iteration, following the simple rule

$$\mathbf{w} = [w_1, \ldots, w_i, \ldots, w_J] = \left[\frac{1}{\sum_{j=1}^{J} j}, \frac{2}{\sum_{j=1}^{J} j}, \ldots, \frac{J}{\sum_{j=1}^{J} j}\right], \tag{5}$$

$$\sum_{i=1}^{J} w_i = \sum_{i=1}^{J} \frac{i}{\sum_{j=1}^{J} j} = \frac{\sum_{i=1}^{J} i}{\sum_{j=1}^{J} j} = 1, \tag{6}$$

where J represents the number of components of GMMs. In this context, fixing the $\mathbf{w}$ will not obviously affect the expressiveness of the GMMs, as expressiveness remains consistent (or could be enhanced) as the number of components increases [61]. A detailed ablation of this design choice is presented in Section 6.3. We fix $\mathbf{w}$ because we find that a Gaussian component's mean and covariance matrix are highly sensitive to its weights. Therefore, fixing the weights helps stabilize the entire learning process. For a similar reason, we apply the same initial parameters θ_o throughout the learning process.

We also set each Gaussian component to be spherical, meaning the covariance matrix can be expressed as $\Sigma = \mathrm{diag}(\sigma)$, where σ is a vector. This method aligns the shapes of x, μ (mean vector), and σ, as x_i^k and μ have the shape $1 \times T$. By setting the Gaussian components to be spherical, the covariance matrix can also be expressed as a $1 \times T$ vector σ, allowing us to treat x, μ, and σ as $1 \times T$ shaped tokens for the input and output of the Transformer, as shown in Fig. 4. In Section 6.5, we further analyze how the covariance design influences the performance of our method. Algorithm 1 shows how each component's μ and σ are computed in Within-domain Tuning. Finally, We propose the following empirical expression to determine z in ECP modeling

$$z = \mathrm{int}\left(e^{\beta n^k}\right), \tag{7}$$

where n^k is the number of ECP samples used in Within-domain Tuning, β is a parameter, which is the result of empirical testing in $\mathcal{S}$, $\mathrm{int}(\cdot)$ means the integer part of a value.

4.2. Knowledge-transfer tuning

This section demonstrates how to use a Transformer to predict $\vec{er}$. We adopt the encoder part of the Transformer for our model. Fig. 5 illustrates how $\vec{er}$ is obtained in the Knowledge-transfer Tuning process.

We use a **[Pad]**[2] token to align the shapes of inputs with varying sizes n^k. The encoder's parameters are updated based on the loss function explained in Section 4.3. $d(\cdot)$, shown in Fig. 5, represents the date information, which indicates the day of the year for x_i, while $p(\cdot)$ represents the parameter information, indicating which Gaussian component of the GMMs μ and σ belong to.

Regarding the design of the encoder, we use RMSNorm, and we do not use Positional Encoding (PE). Considering a set of samples from the kth domain and assuming an ***iid***[3] condition, the order of the samples does not affect the distribution and corresponding parameters of the GMMs. For instance, if $n^k = 20$, incorporating PE would result in $20! \approx 2.43 \times 10^{18}$ possible arrangements, significantly expanding the input space. Omitting PE reduces these 20! cases to 1 case, thus enabling a much more efficient learning process. This is the key reason that motivates us to use the Transformer encoder instead of other DL architectures.

[2] A **[Pad]** token is a special token added to sequences to ensure uniform input sizes when handling data of varying lengths. It aligns shorter sequences by padding them, enabling batch processing, and is ignored by the model during computation.

[3] ***iid*** stands for "independent and identically distributed", meaning samples are independent and drawn from the same probability distribution.

Table 1
Summary of datasets used in experiments.

Data type	Resolution	Amount of domains
ECP-60 min	60 min	17,505 domains (households)
ECP-30 min	30 min	20,737 domains (households)
ECP-15 min	15 min	820 domains (households)
Solar	15 min	73 domains (households)
Transformer-level	15 min	447 domains (substations)

4.3. Loss design

Since GMMs are a white-box model, we can directly compute the negative log-likelihood based on the predicted parameters of the GMMs. The loss is defined as the negative log-likelihood. The loss function is given by

$$\log \mathcal{L}(\mathcal{D}^k \mid \hat{\theta}_r, \mathbf{w}) = -\sum_{i=1}^{N} \log\left(\sum_{j=1}^{J} w_j \mathcal{N}(\mathbf{x}_i^k \mid \mu_j, \mathrm{diag}(\sigma_j))\right), \tag{8}$$

where $w_j \in \mathbf{w}$ is the fixed weight of the jth Gaussian component, N is the number of data points in the domain, J is the number of Gaussian components, $\mathcal{N}(\mathbf{x}_i^k \mid \mu_j, \mathrm{diag}(\sigma_j))$ is the Gaussian probability density function for the jth component with μ_j and σ_j, $\mathcal{D}^k$ is the complete domain dataset.

5. Experiments

5.1. Experiment setting

5.1.1. Data

The data used in Section 5.2 consists of ECP data with 60-minute, 30-minute, and 15-minute resolution from individual households in the UK, Australia, Germany, USA, and the Netherlands. The dataset includes approximately 20 thousand households in total. Due to varying household data lengths, we sample 250 days of ECP data from each household to create a domain. This ensures that each domain has an equal amount of ECP data. To fully utilize all available data, if a household has significantly more than 250 days of data, we sample multiple times, treating each sampled set as an individual domain. We carefully split the data into training, testing, and validation sets with a ratio of 0.8, 0.1, and 0.1, ensuring that the augmented domain remains within the same set. The dataset used in Section 5.3 includes 15-minute resolution data from two sources: transformer-level[4] ECP and solar generation profiles. The transformer-level dataset contains ECP profiles for commercial, residential, and mixed-use buildings (all in transformer-level), while the solar generation dataset provides generation profiles from individual households. Following the same preprocessing strategy, we sample 250 days of data from each source domain to create a consistent domain structure. These datasets are also divided into training, validation, and testing sets using the same 0.8/0.1/0.1 ratio. Table 1 summarizes the datasets used in this study.

5.1.2. Experiment design

As discussed in Section 2.2, applying standard FSL methods as a benchmark in our task is challenging. Upon careful consideration, we identified time-series imputation as the most closely related task to our own [62–64]. There are two differences between standard time-series imputation and our method (1) the amount of data available for modeling in our task is significantly smaller, often less than 10% of the complete dataset, and (2) while time-series imputation requires preserving the temporal order of samples, our method prioritizes modeling the overall distribution of the samples, with their sequential order being less relevant. Given that time-series imputation is the closest task to our own, we aim to provide a clear comparison of the advantages of our proposed methods by selecting TimesNet [62], a popular model for time series imputation, as our baseline for experiments in Sections 5.2 and 5.3. Unlike standard data imputation tasks where the model has access to 80% to 50% of the complete time points, in this task, we provide both our model and TimesNet with only 1.6% to 10% of the complete time points during training. Our primary objective is to model the distribution of ECP data rather than to impute missing data. Therefore, to make the comparison fair, we also adjust TimesNet in this research by employing Maximum Mean Discrepancy (MMD), which is defined in Appendix, as the loss function to train TimesNet instead of Mean Squared Error (MSE) loss used in the original paper [62], MMD is a loss function without considering the sequential order. We employed TimesNet to impute missing data, subsequently dividing the imputed time series into ECP samples to compare their distribution against the domain's complete dataset. Similarly, we generated ECP samples from predicted GMMs using our method to evaluate distribution differences relative to the domain's complete data. Additionally, to more comprehensively demonstrate the advantages of our proposed method in FSL-based ECP distribution modeling, we benchmark it against popular conditional generative models such as VAE, Flow-based model, and Diffusion model in Sections 5.4 and 5.5.

5.1.3. Evaluation methods

Regarding the evaluation metrics, we use MMD, Kullback–Leibler (KL) divergence, Wasserstein distance (WD), MSE of the mean, and Kolmogorov–Smirnov (KS) distance to evaluate the distribution differences, following the methodologies outlined in [8,10]. A smaller value of these metrics indicates better performance. These evaluation metrics are also described in Appendix.

5.1.4. Hyperparameters and training

For the main experiments (detailed in Sections 5.2 and 5.3), we trained our model using an NVIDIA V100 and an NVIDIA A10 GPU. For the training process, we utilize cyclical learning rates [63], with the highest and lowest learning rates set to $1e-3$ and $1e-5$, respectively. The batch size is 128. Both our model and TimesNet have a similar parameter scale, approximately 4.5 million parameters. The number of components of the GMMs, J, is set to 6. The β in Expression (7) set to 0.015. For the source domain, N^{k_s} is 250, while for the target domain, n^{k_t} ranges from 1 to 24 for 60-minute resolution, 1 to 48 for 30-minute resolution, and 1 to 96 for 15-minute resolution. The *Within-domain Tuning*, which follows Algorithm 1, is performed in batch (i.e., in parallel) during training. The z-step is computed according to the expression (7). More detailed experimental settings can be found in our repository.

5.2. Results of residential Electricity Consumption Profiles

In this section, we present and discuss the experimental results of our proposed method applied to data of residential ECP. Fig. 6 presents part of the experimental results on 60-minute resolution ECP data, evaluated across varying values of n^{k_t}—the number of shots or sampled ECP instances provided as model input. From Fig. 6 we can observe that our method effectively captures residential electricity consumption behavior volatility and restores the ECP distribution with minimal input samples in almost all examples. In contrast, TimesNet MSE struggles with volatile time series patterns, primarily generating ECP samples in high-likelihood areas. Although TimesNet with MMD outperforms TimesNet with MSE in generating more volatile patterns, it occasionally fails to accurately capture the ECP distribution. For example, as shown in the first row of Fig. 6(a), TimesNet MMD generates ECPs with smaller peaks. Similarly, in the sixth row of Fig. 6(d), it fails to reproduce the consumption patterns.

[4] To avoid ambiguity between "transformers" in deep learning and in power systems, we use transformer-level to refer specifically to transformers in the power grid.

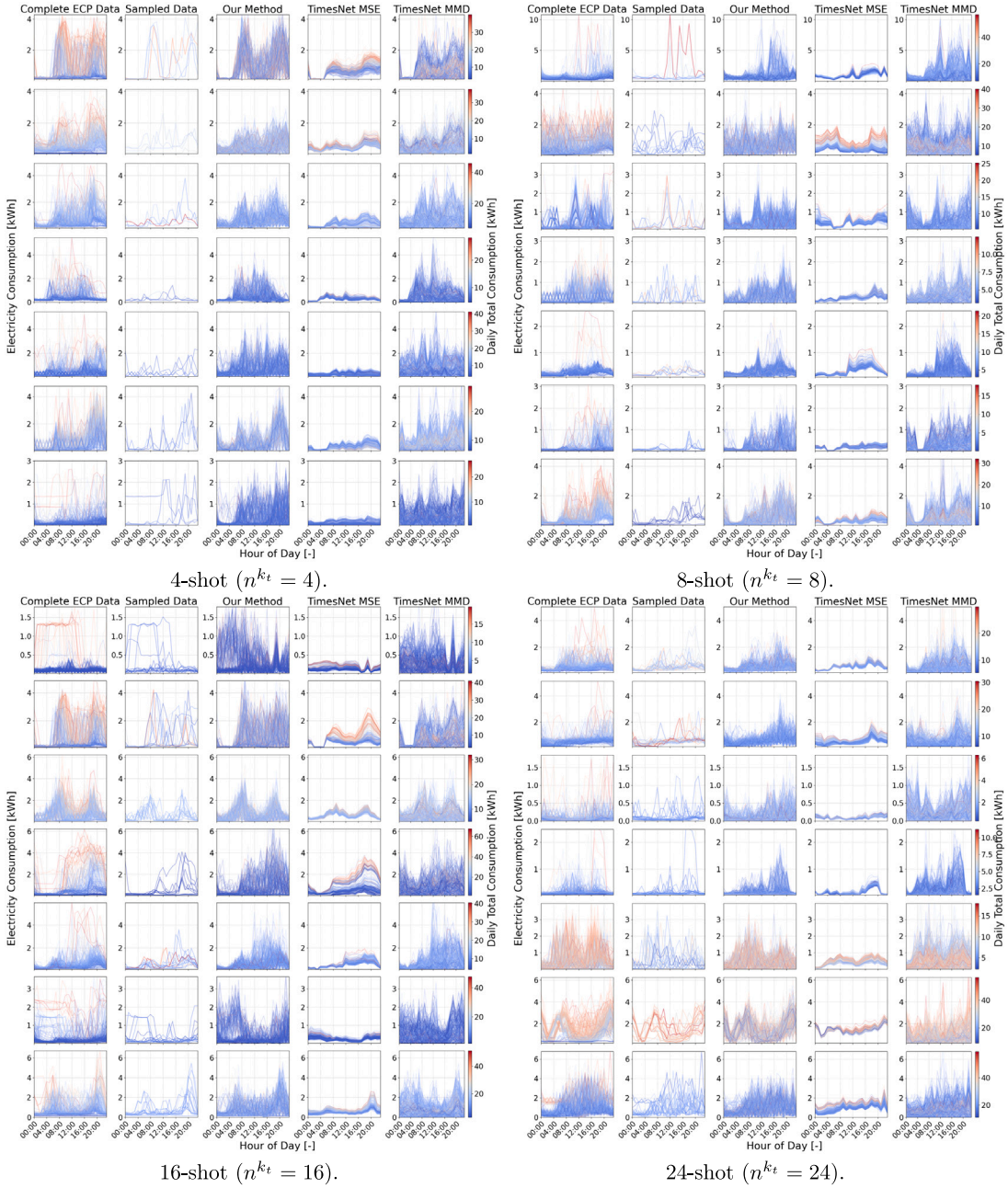


Fig. 6. In each subfigure, every row represents the experimental results for a specific target domain (hourly resolution). Each row, from left to right, includes (1) the complete ECP data of the domain, (2) the sampled ECP data used as input for our model and two TimesNets, (3) the ECP data generated from GMMs whose parameters are predicted by our method, (4) results generated by TimesNet using MSE loss, and (5) results generated by TimesNet using MMD loss. The color of the curves is related to the daily total electricity consumption, more intense red indicates higher daily consumption. n^{k_t} denotes the number of sampled ECPs used for ECP distribution modeling.

Table 2 provides a more quantitative summary of the comparison across 12 experimental scenarios. The results clearly demonstrate that our method consistently achieves lower metric values than the baseline approaches in all cases, with the sole exception being the 32-shot setting at 30-minute resolution, where the sampled real ECP data performs marginally better.

Fig. 7 shows the comparison results of n^{k_t} range from 1 to 24 of 60-minute resolution datasets, where it can be seen that the estimated distribution from our method is closer to the original distribution, especially when the number of shots $1 < n^{k_t} \leq 14$, compared to the sampled ECP data, TimesNet MMD and TimesNet MSE.

5.3. Results of solar and transformer-level profiles

In the previous experiment, we primarily evaluated our method using residential ECP. In this section, we extend the evaluation to the solar and transformer-level datasets.

Fig. 8 presents a subset of the generation results for the solar dataset. It is evident that our proposed method outperforms TimesNet with MMD. TimesNet with MMD demonstrates inferior performance in Fig. 8, which may be attributed to the difficulty of optimizing MMD in high-dimensional settings. Although TimesNet with MSE appears to generate more realistic solar profiles visually, it still fails to accurately

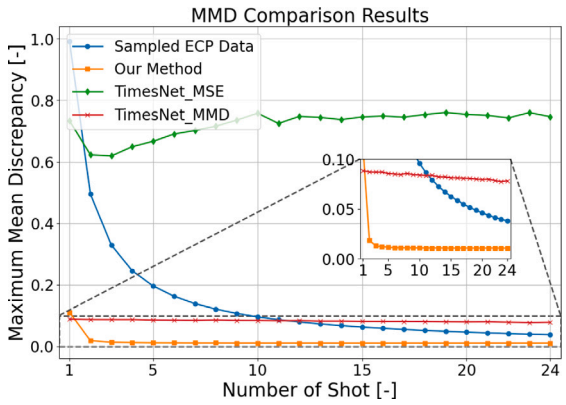


Fig. 7. Comparison of MMD values in 60-minute resolution datasets. For each shot, we sample 100 target domains from $\mathcal{T}$ and compute the average MMD values of n^{k_i} (number of shots) from 1 to 24.

capture the underlying distribution of the original data based on the quantitative comparison of all methods in Table 3.

Fig. 9 presents representative generation results for four types of transformer-level ECP datasets: residential without photovoltaic (PV), residential mixed with commercial, commercial, and residential with PV. As shown in Fig. 9, TimesNet struggles to replicate the original data distribution; specifically, TimesNet with MMD produces overly volatile profiles, while TimesNet with MSE yields less dynamic yet still unrealistic patterns. In contrast, our proposed method exhibits superior robustness and consistently generates realistic samples across all scenarios.

From Table 3, we observe that although our method consistently outperforms the TimesNets on both solar and transformer-level datasets, the performance gap between our method and the sampled ECP is less significant, particularly in the solar dataset. We think this could be attributed to the limited availability of source domain data for these experiments (as summarized in Table 1). For residential ECP, the training set includes close to one thousand to tens of thousands of source domain samples, allowing the model to be fully trained (especially for a data-hungry knowledge-transfer Tuning component). In contrast, the solar and transformer-level datasets contain only several dozen or a few hundred source domain samples, which may limit the model's generalization capability.

5.4. Comparison with conditional generative models

In this section, we compare our proposed method with a conditional VAE (CVAE), Flow-based model (CFlow) [7], and Diffusion model (CDDPM) [10,65] using the 15-minute resolution residential ECP dataset described in Table 1. To ensure consistent experimental settings, we train four separate CVAE models conditioned on 4, 8, 16, and 32 ECP samples, respectively. Each conditional deep generative model is trained to generate a complete ECP dataset of the target domain.

Table 4 presents a summary of evaluation metrics. As shown, our proposed method outperforms the other generative models across all metrics. Fig. 10 provides a qualitative comparison between the generation outputs of the methods. It can be observed that conditional deep generative models exhibit similar behavior to TimesNet with MSE, often failing to capture volatile patterns in the time series. Instead, it tends to generate samples concentrated in high-probability regions.

In this experiment, we want to demonstrate two fundamental weaknesses of CVAE, CFlow, CDDPM or other standard conditional generative frameworks for the proposed generation task in this paper:

- **Fixed-format conditioning:** A conditional deep generative model, for example CVAE, trained on a fixed input shape (e.g., 4-shot with input of shape 4×96) cannot generalize to a different number of input samples (e.g., 8-shot) without retraining a new model. This restricts flexibility and reusability.

Table 2
Results of evaluation metrics of residential ECP.

Method	MMD	KL	KS	WD	MSE.M
4-shot (60-minute resolution)					
Sampled ECP	0.2434	0.9812	0.4244	0.4092	0.0392
TimesNet MSE	0.4938	13.419	0.4307	0.3544	0.0716
TimesNet MMD	0.0868	1.527	0.1912	0.3299	0.0337
Our Method	**0.0222**	**0.4177**	**0.1499**	**0.1530**	**0.0171**
8-shot (60-minute resolution)					
Sampled ECP	0.1196	0.7277	0.3044	0.2999	0.0192
TimesNet MSE	0.6008	13.613	0.4241	0.3639	0.0316
TimesNet MMD	0.0854	1.4935	0.1888	0.3267	0.0332
Our Method	**0.0154**	**0.3535**	**0.1370**	**0.1339**	**0.0099**
16-shot (60-minute resolution)					
Sampled ECP	0.0579	0.4885	0.2353	0.2084	0.0096
TimesNet MSE	0.6656	14.579	0.4366	0.3817	0.0255
TimesNet MMD	0.0809	1.4251	0.1893	0.3228	0.0315
Our Method	**0.0126**	**0.3496**	**0.1315**	**0.1272**	**0.0062**
24-shot (60-minute resolution)					
Sampled ECP	0.0371	0.3935	0.1985	0.1666	0.0058
TimesNet MSE	0.6960	15.332	0.4496	0.3942	0.0246
TimesNet MMD	0.0780	1.3695	0.1876	0.3194	0.0303
Our Method	**0.0116**	**0.3358**	**0.1253**	**0.1217**	**0.0046**
4-shot (30-minute resolution)					
Sampled ECP	0.1246	1.2658	0.4040	0.1460	0.0030
TimesNet MSE	0.2995	1.4999	0.3120	0.2191	0.0130
TimesNet MMD	0.0896	0.5281	0.2448	0.0972	0.0055
Our Method	**0.0607**	**0.4922**	**0.2800**	**0.1214**	**0.0015**
8-shot (30-minute resolution)					
Sampled ECP	0.0570	0.8109	0.3987	0.1318	0.0014
TimesNet MSE	0.2992	1.5017	0.3080	0.2125	0.0131
TimesNet MMD	0.0890	0.4949	0.2279	0.0926	0.0052
Our Method	**0.0367**	**0.3540**	**0.1776**	**0.0746**	**0.0007**
16-shot (30-minute resolution)					
Sampled ECP	0.0324	0.6685	0.3036	0.1004	**0.0008**
TimesNet MSE	0.3003	1.4285	0.3088	0.2103	0.0131
TimesNet MMD	0.0817	0.4887	0.2312	0.0943	0.0048
Our Method	**0.0317**	**0.3957**	**0.1880**	**0.0779**	**0.0008**
32-shot (30-minute resolution)					
Sampled ECP	0.0238	0.4811	**0.2000**	**0.0721**	0.0008
TimesNet MSE	0.3021	1.4789	0.3112	0.2118	0.0132
TimesNet MMD	0.0757	0.5588	0.2352	0.0908	0.0046
Our Method	**0.0284**	**0.3545**	0.2002	0.0747	**0.0006**
4-shot (15-minute resolution)					
Sampled ECP	0.2646	0.2871	0.2477	0.1145	0.0269
TimesNet MSE	0.5603	0.3700	0.3884	0.1285	0.0305
TimesNet MMD	0.2461	13.5407	0.4340	0.3725	0.0078
Our Method	**0.0238**	**0.2132**	**0.2145**	**0.1077**	**0.0019**
8-shot (15-minute resolution)					
Sampled ECP	0.1394	0.2381	0.1920	0.0918	0.0089
TimesNet MSE	0.5743	0.3111	0.3631	0.1267	0.0306
TimesNet MMD	0.2089	13.9127	0.4351	0.3747	0.0085
Our Method	**0.0208**	**0.1730**	**0.1920**	**0.0992**	**0.0015**
16-shot (15-minute resolution)					
Sampled ECP	0.0751	0.1859	0.1441	0.0692	0.0045
TimesNet MSE	0.6201	0.3129	0.2960	0.1323	0.0301
TimesNet MMD	0.1437	15.6473	0.4280	0.3869	0.0083
Our Method	**0.0205**	**0.1232**	**0.1340**	**0.0634**	**0.0016**
32-shot (15-minute resolution)					
Sampled ECP	0.0444	0.1760	0.1378	0.0561	0.0023
TimesNet MSE	0.6451	0.3879	0.3082	0.1406	0.0158
TimesNet MMD	0.1342	15.2995	0.4183	0.4013	0.0130
Our Method	**0.0201**	**0.1227**	**0.1237**	**0.0515**	**0.0015**

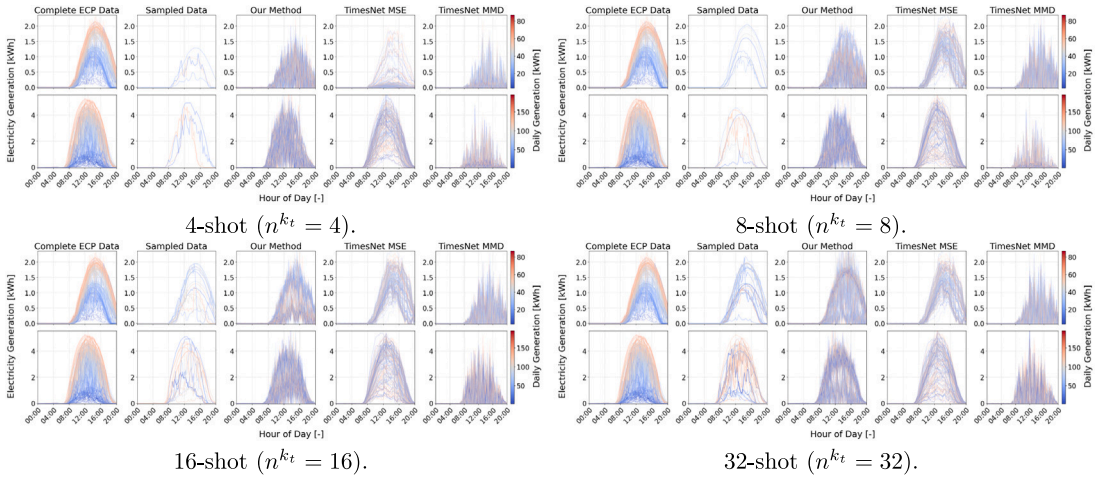


Fig. 8. Similar to Fig. 6, in each subfigure, every row represents the experimental results for a specific target domain of solar generation profile data (15-minute resolution). Each row, from left to right, shows: (1) the complete solar profile of the domain, (2) the sampled profiles used as input for our method and both TimesNet models, (3) the profiles generated from GMMs with parameters predicted by our method, (4) TimesNet results trained with MSE loss, and (5) TimesNet results trained with MMD loss. Curve color indicates total daily solar generation, where more intense red represents higher output. n^{k_t} denotes the number of sampled profiles used for distribution modeling.

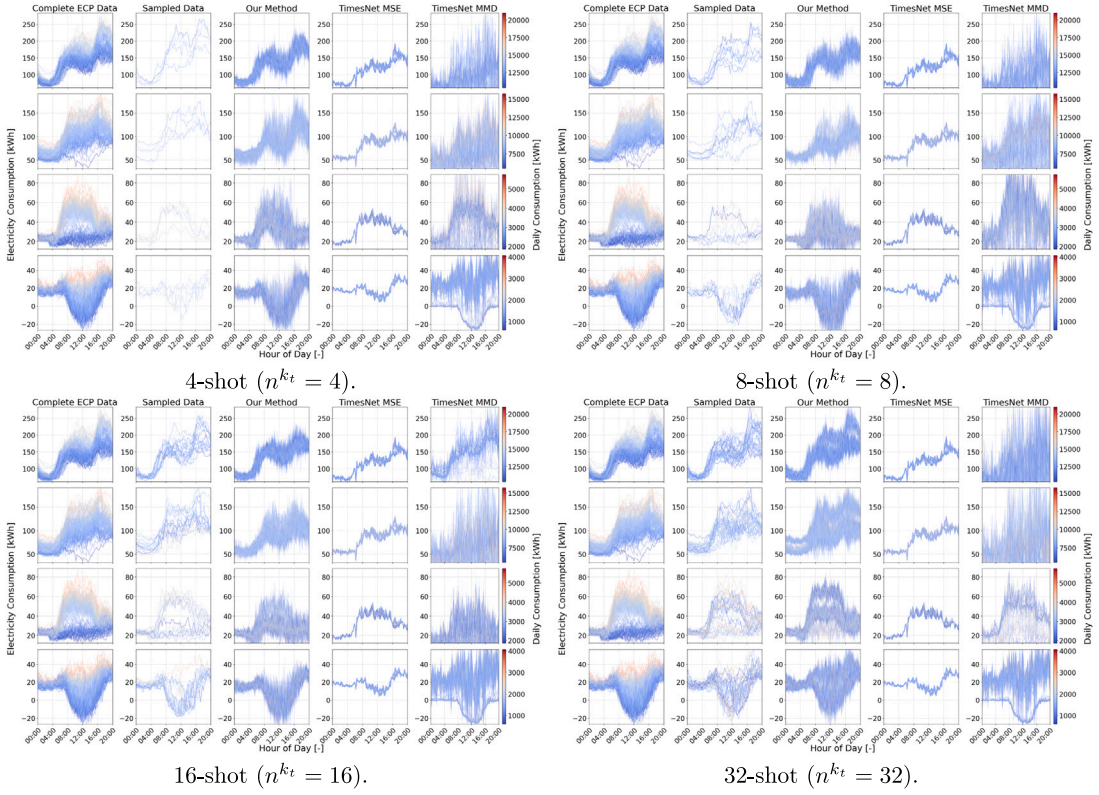


Fig. 9. Similar to Fig. 6, in each subfigure, every row represents the experimental results for a specific target domain of transformer-level ECP data (15-minute resolution). In each subfigure, Row 1 shows transformer-level residential ECP without PV, Row 2 shows transformer-level residential mixed with commercial ECP, Row 3 shows transformer-level commercial ECP, and Row 4 shows transformer-level residential ECP with PV. Each row compares the results of our method with the benchmark models. Curve color reflects the total daily electricity consumption, where more intense red indicates higher consumption. n^{k_t} denotes the number of sampled profiles used for distribution modeling.

- **Order sensitivity:** The condition input is usually treated as a matrix or tensor, meaning different permutations of the same set of input samples result in different conditions. We conceptually show the permutation cases of input tensors/matrixs in Fig. 11. For instance, with 32 ECP samples (32-shot), there are $32! \approx 8.2 \times 10^{33}$ possible orderings, all representing essentially the same information. Learning under such redundancy adds significant complexity and degrades performance.

As illustrated in Fig. 11, our method avoids these pitfalls by design. In the *Within-domain Tuning* stage, our method resembles an EM algorithm, which is inherently *order-invariant*. Furthermore, in the

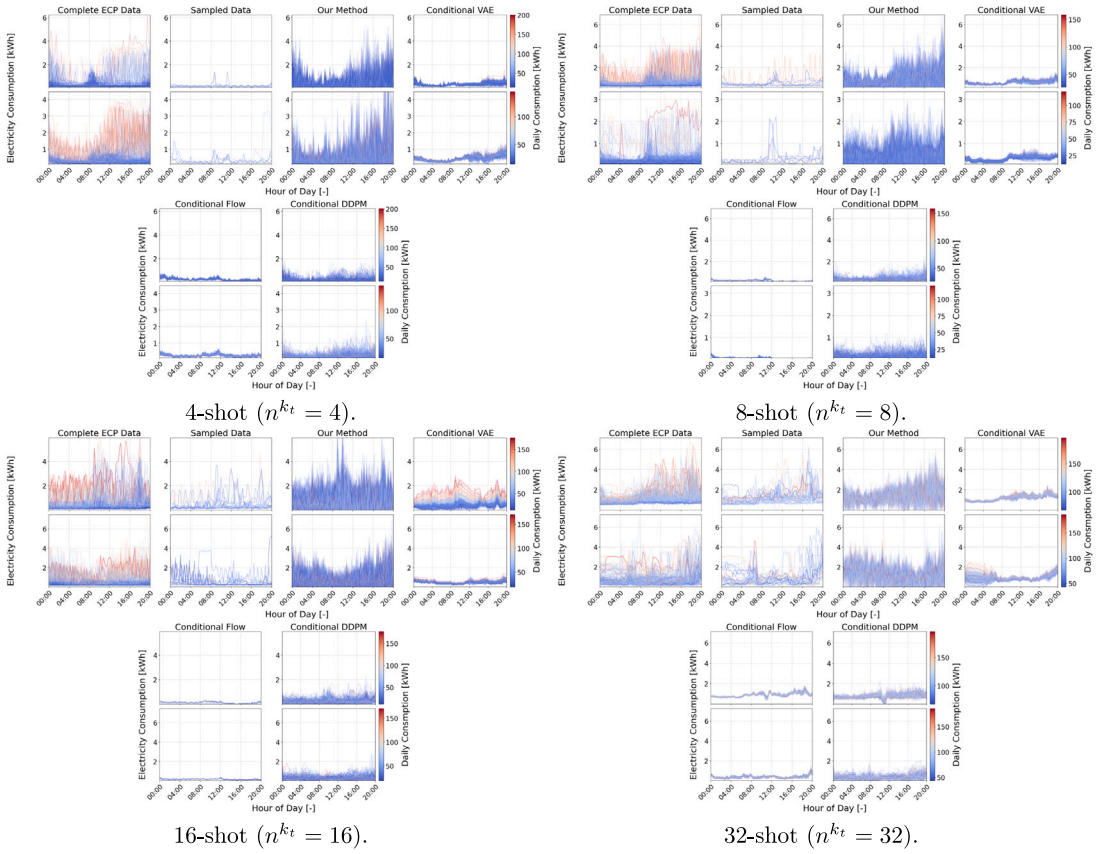


Fig. 10. The generated results from our proposed method, CVAE, CFlow, and CDDPM. Each subfigure presents the experimental results of two target domains. For fair comparison, we use the same input conditions and the model trained in Section 5.2. Similar to Fig. 6, curve color reflects the total daily electricity consumption, where more intense red indicates higher demand. n^{k_t} denotes the number of sampled profiles used for distribution modeling.

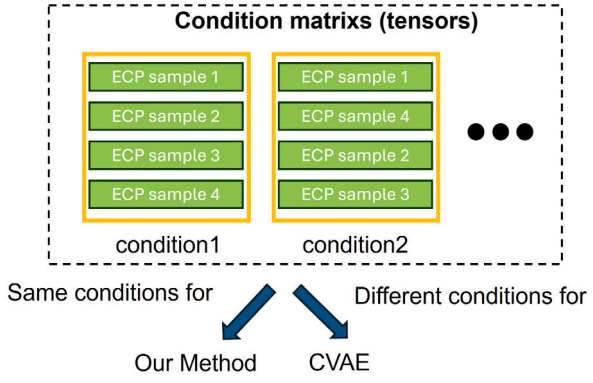


Fig. 11. Illustration of order sensitivity in CVAE. Although the two input sets contain the same ECP samples in different orders (Condition 1 vs. Condition 2), CVAE treats them as entirely different conditions. In contrast, our method is order-invariant, interpreting all permutations of the same samples as a single condition and producing consistent generations.

Knowledge-transfer Tuning stage (see Section 4.2), we explicitly remove PE from the transformer to make the model *order-invariant*. As a result of this design, in the case of 32-shot generation, where a conventional conditional model must account for up to $32! \approx 8.2 \times 10^{33}$ possible orderings of the conditioning set, our method reduces this to 1 set.

In real-world scenarios, there is no explicit indexing of samples (e.g., ECP sample 1, ECP sample 2) as depicted in Fig. 11. Instead, each ECP sample i can be understood as representing a particular type or pattern of electricity consumption. The ordering of these patterns introduces redundant information when used as conditions in a deep generative model framework (while these different patterns are from one domain). Our architectural choice ensures that our model generalizes well regardless of the number or order of conditional samples, which explains its superior performance compared to deep generative models as benchmarks.

5.5. Comparison with classical fine-tuning

In this section, we compare our method with a classical fine-tuning approach, in which a generative model is first trained on the source domains and then fine-tuned on one target domain dataset [17,49,50]. For this experiment, we use an 8-shot CVAE model (as described in Section 5.4, using 8 ECP samples as condition), fine-tuned using a small learning rate (lr = 0.00005). The 15-minute resolution ECP dataset is used, and the experiment is conducted using an NVIDIA A10 GPU.

Fig. 12 illustrates how the generated ECPs evolve throughout the fine-tuning process, gradually aligning with the distribution of the original data. Fig. 13 shows how the MMD loss decreases over training steps and elapsed time. Notably, although the CVAE surpasses our method in performance after approximately 92 s of fine-tuning, our method reaches an equivalent MMD of 0.0198 in only 0.021 s.

It is important to note that in this experiment, the batch size for our method is set to 1. In practice, however, our approach can easily leverage larger batch sizes for parallel processing. In contrast, parallelizing fine-tuning across thousands of domains (i.e., thousands of DL-based generative models need to be trained and stored) is significantly more complex and even impractical.

This result supports the argument made in Section 1 that one of the key contributions of our proposed method is its efficiency.

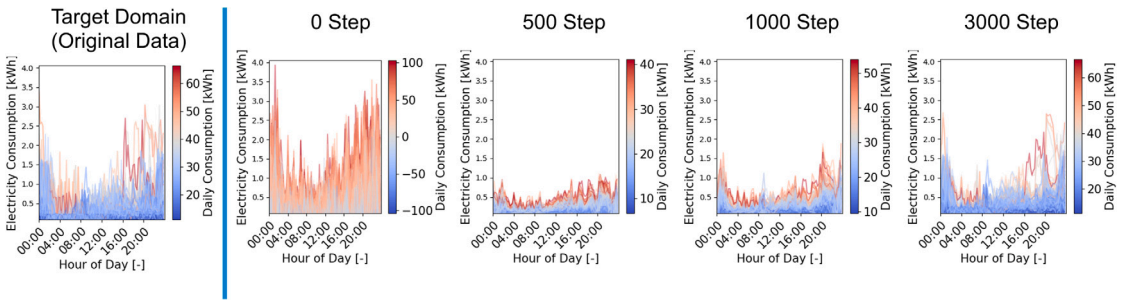


Fig. 12. The generation results of a CVAE model fine-tuned for a specific target domain. From left to right, it displays the original ECP data followed by the CVAE-generated outputs after 0, 500, 1000, and 3000 fine-tuning steps.

Table 3
Results of evaluation metrics of solar and transformer data.

Method	MMD	KL	KS	WD	MSE.M
4-shot (Solar)					
Sampled ECP	0.1529	**1.3643**	0.5253	**0.1180**	0.0033
TimesNet MSE	0.5086	2.4618	0.6487	0.5039	0.0013
TimesNet MMD	0.4623	3.3739	0.7329	0.5806	0.0013
Our Method	**0.0248**	2.1398	**0.3101**	0.3930	**0.0007**
8-shot (Solar)					
Sampled ECP	0.0722	**0.9049**	0.4161	**0.0746**	0.0012
TimesNet MSE	0.5311	2.1840	0.6188	0.5227	0.0013
TimesNet MMD	0.4719	3.5722	0.7270	0.5869	0.0013
Our Method	**0.0241**	1.7944	**0.3119**	0.3926	**0.0006**
16-shot (Solar)					
Sampled ECP	0.0357	**0.3924**	0.308	**0.0468**	0.0008
TimesNet MSE	0.5318	2.1867	0.6157	0.5219	0.0013
TimesNet MMD	0.4592	3.4792	0.7193	0.5886	0.0013
Our Method	**0.0237**	1.6541	**0.3127**	0.3851	**0.0006**
32-shot (Solar)					
Sampled ECP	0.0232	**0.2931**	**0.1995**	**0.0314**	**0.0003**
TimesNet MSE	0.5319	2.1879	0.6160	0.5183	0.0012
TimesNet MMD	0.4613	3.4200	0.7140	0.5942	0.0013
Our Method	**0.0229**	1.2608	0.3136	0.3746	**0.0003**
4-shot (Transformer-level)					
Sampled ECP	0.2458	0.7612	0.2660	7.4889	0.0868
TimesNet MSE	0.5142	4.9043	0.3650	10.7955	0.2863
TimesNet MMD	0.1723	4.1651	0.2992	15.3859	0.0434
Our Method	**0.0668**	**0.6101**	**0.1504**	**7.0274**	**0.0481**
8-shot (Transformer-level)					
Sampled ECP	0.1178	0.5991	0.2142	2.5204	0.0268
TimesNet MSE	0.5198	4.9047	0.3653	10.8114	0.2866
TimesNet MMD	0.1333	4.2887	0.2836	16.1775	0.0255
Our Method	**0.0493**	**0.5210**	**0.1045**	**5.1290**	**0.0109**
16-shot (transformer-level)					
Sampled ECP	0.0584	0.5307	0.1112	3.2449	0.0079
TimesNet MSE	0.5188	4.9072	0.3656	10.8218	0.2857
TimesNet MMD	0.1289	4.0919	0.3171	16.4256	0.0304
Our Method	**0.0402**	**0.5101**	**0.0556**	**2.2257**	**0.0018**
32-shot (Transformer-level)					
Sampled ECP	**0.0272**	**0.4918**	0.0527	**1.4543**	0.0042
TimesNet MSE	0.5184	4.9146	0.3666	10.9160	0.1288
TimesNet MMD	0.1224	4.3801	0.4293	22.7491	0.0361
Our Method	0.0398	0.4998	**0.0479**	2.1282	**0.0015**

Table 4
Evaluation metrics for CVAE vs. Our method vs. DDPM/Flow on 15-minute resolution residential ECP.

Method	MMD	KL	KS	WD	MSE.M
4-shot					
CVAE	0.2868	0.3414	0.2441	0.2618	0.0761
Our Method	**0.0238**	**0.2132**	**0.2145**	**0.1077**	**0.0019**
CDDPM	0.1760	0.2911	0.4106	0.5484	0.0567
CFlow	0.4949	0.3684	0.2566	0.3906	0.0714
8-shot					
CVAE	0.4089	0.3222	0.2888	0.2337	0.0528
Our Method	**0.0208**	**0.1730**	**0.1920**	**0.0992**	**0.0015**
CDDPM	0.1234	0.3048	0.3319	0.3680	0.0607
CFlow	0.8356	0.3772	0.6516	0.5345	0.0675
16-shot					
CVAE	0.2443	0.3132	0.2071	0.2514	0.0809
Our Method	**0.0205**	**0.1232**	**0.1340**	**0.0634**	**0.0016**
CDDPM	0.0905	0.2883	0.3358	0.5902	0.0783
CFlow	0.4427	0.3635	0.4147	0.7274	0.0930
32-shot					
CVAE	0.3167	0.3078	0.2330	0.2809	0.0740
Our Method	**0.0201**	**0.1227**	**0.1237**	**0.0515**	**0.0015**
CDDPM	0.1539	0.2956	0.3477	0.5230	0.0578
CFlow	0.4522	0.3719	0.2145	0.3033	0.0650

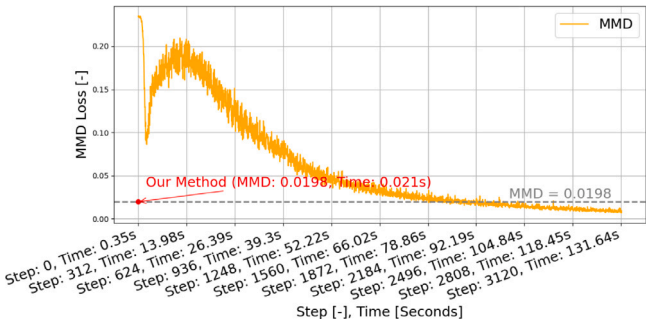


Fig. 13. MMD over training time during fine-tuning. The orange curve shows the evolution of the MMD as training progresses for CVAE. The x-axis indicates both the training step and the corresponding training time. A dashed horizontal line marks the MMD value of 0.0198 for reference. The red point and annotation highlight the performance of Our Method, which achieves an MMD of 0.0198 at just 0.021 s.

By replacing conventional fine-tuning with batch EM, our framework enables straightforward parallel learning of domain-specific knowledge using PyTorch, making it a more practical and scalable solution in multi-domain settings.

5.6. Interpretability analysis

In this section, we analyze how our method models the ECP of a target domain, focusing on the *Knowledge-transfer Tuning*. Our goal is to quantitatively and empirically assess its contribution to predicting the vector $\vec{er}$. While *Within-domain Tuning* is an interpretable EM algorithm for GMMs, *Knowledge-transfer Tuning* acts as a black-box. We therefore focus on understanding its role and impact on the method's behavior.

Inspired by interpretability tools such as attention maps [66] and feature importance measures [67], we measure the influence of each individual input ECP sample on the output of *Knowledge-transfer Tuning*

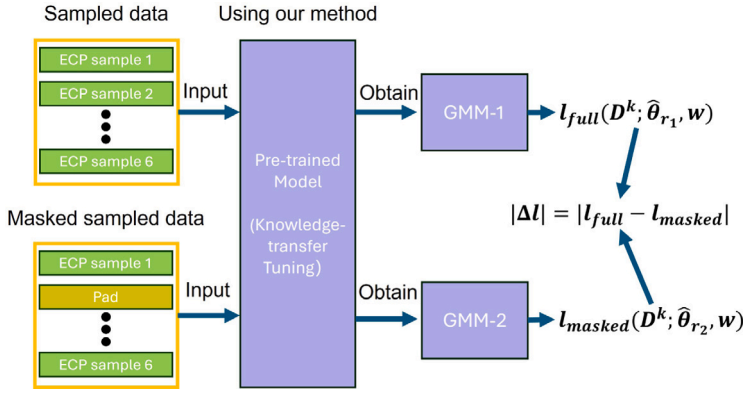
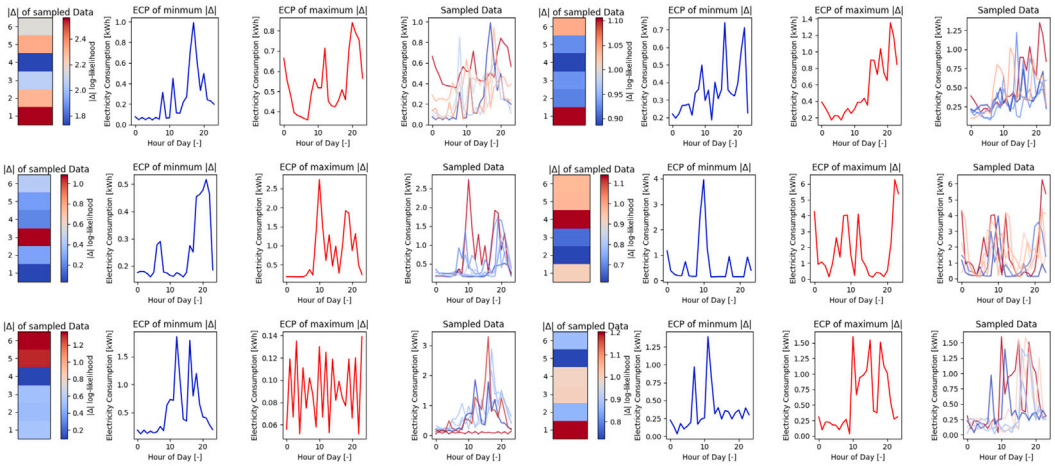


Fig. 14. Illustration of the $|\Delta\mathcal{L}|$ computation for interpretability. Given a set of sampled ECPs (top left), each profile is sequentially masked (bottom left) and fed into the pre-trained transformer model. The model outputs (parameters of) two GMMs (GMM-1, GMM-2), from which we compute the negative log-likelihood using Eq. (8). The absolute difference between these two likelihoods ($|\Delta\mathcal{L}|$) quantifies the effect of each individual ECP profile on the predicted vector $\vec{er}$.



Cases with relatively large variance of $|\Delta\mathcal{L}|$. Cases with relatively small variance of $|\Delta\mathcal{L}|$.

Fig. 15. Visualization of the interpretability experiment measuring $|\Delta\mathcal{L}|$ for individual ECP samples. As depicted in Fig. 14, we sample six ECP samples and mask one of them to compute $|\Delta\mathcal{L}|$ (or $|\Delta|$ for simplicity). The heatmap on the left shows the change in $|\Delta\mathcal{L}|$ when masking each sample, with higher values indicating greater influence on the model output. The second and third columns display the specific ECP samples with the smallest and largest $|\Delta\mathcal{L}|$, respectively. The rightmost column shows all sampled ECPs colored by their corresponding $|\Delta\mathcal{L}|$ in the heatmap.

using a similar way. Concretely, for each ECP sample in the input, we mask it out, recompute the negative log-likelihood $\mathcal{L}$ (using Eq. (8)), and record the change

$$|\Delta\mathcal{L}| = |\mathcal{L}_{\text{full}} - \mathcal{L}_{\text{masked}}|. \tag{9}$$

This $|\Delta\mathcal{L}|$ (or $|\Delta|$ for simplification in Fig. 15) quantifies how strongly each sample affects predicting the vector $\vec{er}$, as a larger $|\Delta\mathcal{L}|$ indicating a larger effect of the sample on predicting the $\vec{er}$. Fig. 14 illustrates the procedure for computing $\Delta\mathcal{L}$ in our experiments.

We use the 60-minute resolution ECP dataset, as described in Table 1, for this experiment. Fig. 15 reveals how the pattern of each sampled ECP influences the overall modeling quality via $|\Delta\mathcal{L}|$. We observe that when an individual ECP sample exhibits a pattern that is substantially different from the rest of the sampled set, its removal results in a large difference in negative log-likelihood loss (i.e., high $|\Delta\mathcal{L}|$). For example, on the left side of Fig. 15, we present examples characterized by relatively high variance in $|\Delta\mathcal{L}|$. In these cases, the ECP sample associated with the largest $|\Delta\mathcal{L}|$ typically deviates obviously from the other samples, either in magnitude (e.g., exceptionally high or low values) or temporal structure. Conversely, the right side of Fig. 15 shows cases with relatively low variance in $|\Delta\mathcal{L}|$. Here, even the sample with the highest $|\Delta\mathcal{L}|$ tends to exhibit a pattern similar to the remaining samples, resulting in a smaller change in the $|\Delta\mathcal{L}|$ when masked.

The analysis suggests that input ECP samples that are redundant or similar to others have less impact on the model's prediction, as indicated by smaller changes in $|\Delta\mathcal{L}|$ when removed. In contrast, unique or highly informative samples have a larger impact.

6. Ablation study

6.1. Within-domain tuning and knowledge-transfer tuning

In this section, we evaluate the effectiveness of *Within-domain Tuning* and *Knowledge-transfer Tuning* on our method's performance. To assess *Within-domain Tuning*, we conducted an experiment omitting this component, where the Transformer encoder alone predicted $\vec{or}$, followed by $\hat{\theta}_r = \vec{or} + \theta_o$. For *Knowledge-transfer Tuning*, we examined the effect of excluding the encoder's PE by training a model with traditional absolute PE in the encoder [22]. All ablation experiments were conducted at a similar scale of parameters, using compact models with approximately 35,000 parameters. Remarkably, our method proved to be still effective even with this reduced model size.

Fig. 16 shows MMD values for n^{k_t} ranging from 1 to 24 and 1 to 48, comparing performance with and without *Within-domain Tuning* at hourly and half-hourly resolutions. The results indicate performance

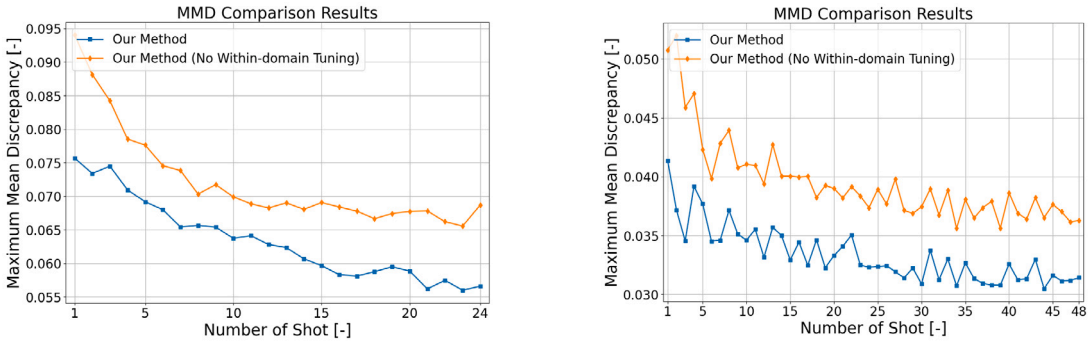


Fig. 16. Comparison of with and without Within-domain Tuning.

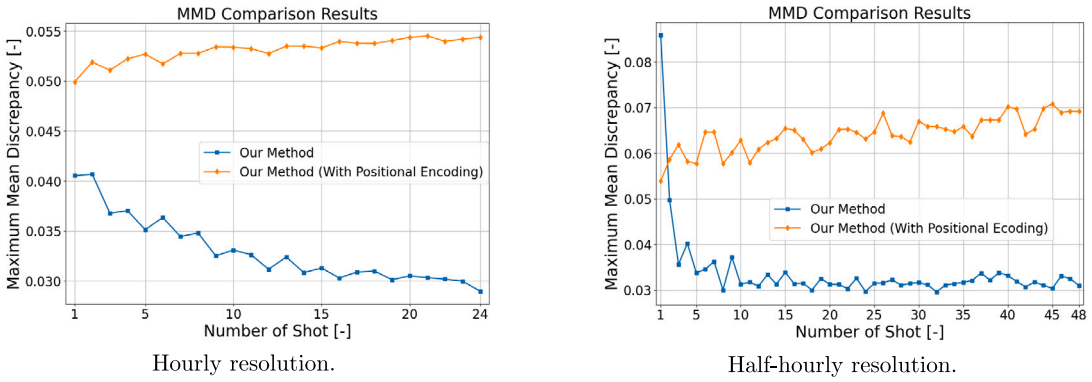


Fig. 17. Comparison of without and with PE.

degradation without *Within-domain Tuning*, likely due to the substantial parameter-space distance $\|\theta_o - \theta_r\|$ in GMMs, as discussed in Section 4.

Similarly, Fig. 17 displays MMD values across shot counts, comparing models with and without PE. The comparison shows that PE introduces noise, reducing model accuracy. As noted in Section 4.2, each ECP sample is treated as an independent, identical token in GMMs, and PE disrupts this independence, thus hurting performance.

6.2. Quantifying the contribution of within-domain tuning and knowledge-transfer tuning

In the previous section, we demonstrated how *Within-domain Tuning* and *Knowledge-transfer Tuning* each improve the final result, but we have not yet provided a quantitative evaluation of their respective contributions to the loss in Eq. (8). Inspired by Shapley values from game theory [68–70], we treat *Within-domain Tuning* (player 1) and *Knowledge-transfer Tuning* (player 2) as two "players" in a collaborative game. Let $v(S)$ denote the log-likelihood when only the set of players $S \subseteq \{1,2\}$ is applied. The Shapley value φ_i for each player i is then

$$\varphi_i = \sum_{S \subseteq \{1,2\} \setminus \{i\}} \frac{|S|!\,(2-|S|-1)!}{2!} \big[v(S \cup \{i\}) - v(S)\big]. \tag{10}$$

Since there are only two players, this simplifies to

$$\varphi_1 = \tfrac{1}{2}\big[v(\{1\}) - v(\emptyset)\big] + \tfrac{1}{2}\big[v(\{1,2\}) - v(\{2\})\big], \tag{11}$$

$$\varphi_2 = \tfrac{1}{2}\big[v(\{2\}) - v(\emptyset)\big] + \tfrac{1}{2}\big[v(\{1,2\}) - v(\{1\})\big]. \tag{12}$$

Where, $v(\{1,2\})$ is the full model value including both tuning stages, $v(\{1\})$ (resp. $v(\{2\})$) is the loss using only *Within-domain Tuning* (resp. *Knowledge-transfer Tuning*), and $v(\emptyset)$ is the loss with neither component, We follow common practice in Shapley value estimation and assume $v(\emptyset) = 0$ in computation. The values φ_1 and φ_2 thus quantify the average marginal contribution of each tuning stage to the final log-likelihood.

In practice, evaluating all coalitions is trivial for two players, but we may wish to average contributions over many target domains. We therefore approximate the empirical Shapley value [71] by sampling M random coalitions $S^{(1)}, \ldots, S^{(M)} \subseteq \{1,2\} \setminus \{i\}$ across domains and computing

$$\hat{\varphi}_i = \frac{1}{M} \sum_{m=1}^{M} \Big[v\big(S^{(m)} \cup \{i\}\big) - v\big(S^{(m)}\big)\Big]. \tag{13}$$

We use the transformer-level dataset (described in Table 1) for the experiments in this section. Specifically, we sample $M = 2{,}000$ coalitions[5] from the target domains to compute the empirical Shapley values.

Table 5 presents the results. A higher Shapley value indicates a greater contribution of the corresponding player to the overall model performance. Interestingly, we observe that as the number of shots (i.e., sampled ECPs) increases, the contribution of *Within-domain Tuning* ($\hat{\varphi}_1$) also increases. In parallel, the full model value ($v(\{1,2\})$) improves as well. This trend may suggest that the effectiveness of the EM algorithm, which is essentially the *Within-domain Tuning*, improves with more available data, thereby yielding greater contribution to the final estimation.

6.3. Affects of the number of Gaussian components

In the previous sections, we primarily evaluated the impact of *Within-domain Tuning* and *Knowledge-transfer Tuning* on the performance of our proposed method. In this section, we shift our focus to

[5] *Note:* a coalition refers to a subset of players (not domains) under a specific game environment (sampled data from the domain), thus, a single domain can be used multiple times for sampling coalitions.

Table 5
Empirical Shapley value by number of shots.

Number of Shots	$\hat{\varphi}_1$ (Player 1)	$\hat{\varphi}_2$ (Player 2)	$v(\{1,2\})$ (Both)
4-shot	3.8329	58.9010	62.7339
8-shot	13.0924	53.5519	66.6443
16-shot	31.3918	36.0309	67.4227
32-shot	36.4258	34.0081	70.4339

Note: $\hat{\varphi}_1$, $\hat{\varphi}_2$, and $v(\{1,2\})$ denote the Shapley (i.e., log-likelihood) value of *Within-domain Tuning*, *Knowledge-transfer Tuning*, and their joint coalition, respectively.

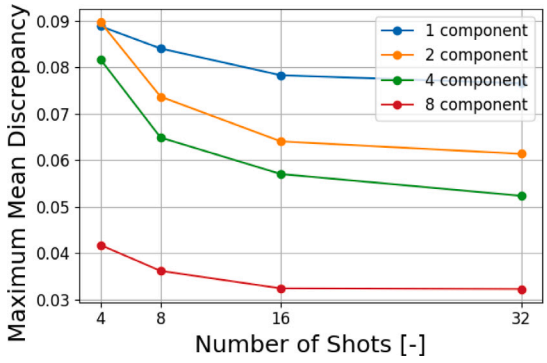


Fig. 18. MMD values of the proposed method evaluated with 1, 2, 4, and 8 Gaussian components across 4, 8, 16, and 32 shots.

investigating how the number of Gaussian components in the GMM affects the overall performance.

This experiment is conducted on the transformer-level dataset, as described in Table 1. To ensure a fair comparison, we train models with varying numbers of Gaussian components—specifically 1, 2, 4, and 8, while maintaining approximately the same total number of parameters (around 4.5 million). This allows us to isolate the effect of the number of components on the model's performance. For each configuration, we sample across all target domains once and compute the average MMD to evaluate performance.

Fig. 18 presents the MMD values for the proposed method across different component settings and shot numbers (4, 8, 16, and 32). As observed in Fig. 18, MMD decreases consistently with an increase in both the number of Gaussian components and the number of shots. This trend suggests that increasing the number of components enhances the expressiveness of the GMMs, enabling it to capture more complex time-series patterns and thereby improving the quality of the generated ECP data.

6.4. Effects of Gaussian component weights

In Section 4.1, we manually fixed the Gaussian mixture weights using Expression (5). However, the choice of mixture weights may influence the generative performance of the model. In this section, we investigate how different weight configurations affect the results.

The experiment is conducted on the transformer-level dataset described in Table 1. To reduce computational cost, we train a series of smaller models, each containing approximately 450 K parameters. Four different weight settings are compared, (1) fixed weights determined by Eq. (5), (2) learnable weights treated as trainable parameters of the model, and (3) two configurations of randomly initialized fixed weights.[6] The number of Gaussian components is fixed to 4 for all experiments.

[6] The two sets of random weights are initialized as $\{0.29, 0.27, 0.31, 0.13\}$ and $\{0.09, 0.27, 0.41, 0.23\}$, rounded to two decimal places.

Table 6
Evaluation metrics for GMM with different weight settings.

Method	MMD	KL	KS	WS	MSE.M
4-shot					
Fixed Weights	0.0742	0.6925	**0.2124**	8.4516	0.0703
Learnable Weight	0.0794	**0.6255**	0.2247	**7.2900**	0.0678
Random Weights 1	0.0870	0.7800	0.2311	8.4396	0.0808
Random Weights 2	**0.0719**	0.7445	0.2286	7.6008	**0.0654**
8-shot					
Fixed Weights	**0.0665**	0.6000	0.1580	6.3096	0.0450
Learnable Weight	0.0678	0.5815	**0.1575**	**5.7924**	0.0433
Random Weights 1	0.0864	**0.5770**	0.1696	6.1044	0.0584
Random Weights 2	0.0679	0.6280	0.1775	6.4868	**0.0405**
16-shot					
Fixed Weights	0.0516	0.5740	0.1002	3.2734	0.0164
Learnable Weight	0.0475	**0.5556**	0.1180	**3.2092**	0.0109
Random Weights 1	0.0626	0.5715	0.1085	3.3307	0.0166
Random Weights 2	**0.0466**	0.5673	**0.0971**	3.3450	**0.0103**
32-shot					
Fixed Weights	0.0452	**0.0519**	0.0582	**2.1822**	0.0029
Learnable Weight	**0.0420**	0.0522	0.6019	2.2137	0.0037
Random Weights 1	0.0585	0.0538	**0.6056**	2.2086	0.0033
Random Weights 2	0.0564	0.0595	0.1637	2.2672	**0.0028**

Table 6 reports the performance of different weight configurations. From these results, we observe that under the same number of components and the same ECP modeling context, the choice of weight setting does not lead to substantial performance differences. Though, theoretically, the learnable weight (weight setting two) should provide more flexibility to the method. We think a possible explanation is similar to the design of diffusion models [72], in which the noise-adding schedule is typically fixed, and making it learnable does not consistently improve generation quality. In our case, a slightly different weight assignment may induce large changes in the mean and covariance of each Gaussian component. Consequently, the additional flexibility of learnable weights may be offset by the instability it introduces, which disrupts the optimization of means and covariances during training.

6.5. Effects of covariance design

In Section 4.1, we assumed a diagonal covariance matrix for each Gaussian component. While computationally efficient, this assumption may restrict the expressiveness of the model. In this section, we investigate whether introducing additional flexibility in covariance modeling can improve generative performance.

Directly learning a full covariance matrix would lead to a large number of trainable parameters and deviates significantly from our initial model design. Instead, we introduce a structured and scalable parameterization that allows us to control the degree of covariance flexibility. In the original setting, we learn a vector σ and define the covariance of each Gaussian component as $\Sigma = \mathrm{diag}(\sigma)$. Here, we extend this formulation by parameterizing the covariance using a low-rank approximation [73] as

$$\Sigma = U^\top U + aI, \tag{14}$$

where $\Sigma \in \mathbb{R}^{T\times T}$, $a > 0$ is a small scalar, and $U \in \mathbb{R}^{rank\times T}$ with $U = \begin{bmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_{\text{rank}} \end{bmatrix}$ representing a set of learned basis vectors. By increasing the *rank*, we can gradually enhance the flexibility of the covariance model without directly learning a full $T \times T$ matrix (which corresponds to $rank = T$), thereby enabling finer–grained control and analysis of covariance structure.

The experiment is conducted on the transformer-level dataset described in Table 1. We train a series of models with different rank values

Table 7
Evaluation metrics for GMM with different ranks.

Method	MMD	KL	KS	WD	MSE.M
4-shot					
Original Setting	**0.0681**	0.6335	**0.1566**	**7.3467**	**0.0497**
Rank 2	0.0957	**0.6229**	0.2864	8.6191	0.0611
Rank 3	0.1091	0.7148	0.3254	9.7281	0.0679
Rank 5	0.1101	0.7758	0.3092	10.6939	0.0695
Rank 10	0.1149	0.8070	0.3187	10.7284	0.0709
32-shot					
Original Setting	**0.0439**	**0.5886**	**0.1086**	**5.2426**	**0.0103**
Rank 2	0.0664	0.6623	0.2070	6.2439	0.0303
Rank 3	0.0653	0.6546	0.2082	6.2404	0.0241
Rank 5	0.0692	0.7033	0.2687	6.5019	0.0321
Rank 10	0.0713	0.7203	0.2723	6.9416	0.0573

of approximately the same amount of paramters (around 1.8 million) and test the 4-shot and 32-shot scenarios.

Table 7 presents the evaluation metrics for GMMs with different covariance ranks. Interestingly, the original configuration (diagonal covariance) consistently yields the best overall performance in both 4-shot and 32-shot experiments. This result may seem counterintuitive at first, as one might expect models with greater expressiveness to better match the distribution. However, we attribute this outcome to two main factors. First, the nature of the problem setting. The target domain is data-limited, requiring the model to perform inference under few-shot conditions. For example, in the 4-shot case, only $4 \times 96 = 384$ observed values are available at inference time. In contrast, a GMM with k components requires predicting $k \times (96 + \text{rank} \times 96)$ parameters, meaning the parameter space expands with the covariance rank. When the number of learnable parameters largely exceeds the number of available observations, the model becomes more prone to unstable. Second, challenges in *Within-Domain Tuning*. Similarly, the EM algorithm must estimate a significantly larger number of parameters with limited data, potentially decreasing the robustness of the *Within-Domain Tuning* process. In such circumstances, a simpler covariance structure – such as the diagonal form – provides better generalization, particularly in the context of residential ECP modeling in this study.

7. Discussion

In Sections 5.2 and 5.3, we compare the proposed FSL method with several benchmarks. We show that the proposed method has better performance than other benchmarks. However, We observe that the superiority of our method is much more pronounced in the experiments of Section 5.2, whereas the improvements in Section 5.3 are comparatively weaker. This discrepancy may be primarily explained by the amount of source-domain knowledge that can be transferred. In Section 5.2, the transformer is trained on thousands to tens of thousands of source domains, enabling it to acquire rich and diverse domain knowledge. In contrast, in Section 5.3, only a few hundred source domains are available. Such a limited source-domain dataset constrains the diversity of learned representations and thus leads to a relatively smaller performance gain. A potential solution to this limitation is to train a foundation model using external data sources, thereby allowing the model to learn a broader spectrum of domain-invariant patterns. For example, acquiring additional historical solar datasets and pre-training the transformer on them may help extract universal characteristics of solar generation, which can then be transferred effectively to the target domains. However, we think it is crucial that the external datasets share common statistical patterns with the target domains, otherwise, the transferred knowledge may be irrelevant or even detrimental to the final performance due to negative transfer.

Additionally, in Section 5.2 and Fig. 7, we show that when a small number of samples are available (e.g., $n^{k_t} > 1$), the MMD of the modeled ECP distribution is significantly closer to the ground truth than the benchmarks (TimesNets). However, we did not explicitly explain why the performance remains less effective when only a single sample is available (i.e., $n^{k_t} = 1$). We think this is because, under extreme data scarcity, it becomes difficult for both steps – *Within-domain Tuning* and *Knowledge-transfer Tuning* – to infer a representative pattern of the target domain. Due to the high variability of residential ECP data, a single daily profile can correspond to many different underlying consumption behaviors. As discussed in Section 6.2 and illustrated in Fig. 15, some ECP samples are more informative and distinguishable than others, contributing more to the predicted GMM parameters. When only one sample is available, the probability of observing such an informative profile decreases, making it challenging for the model to infer the underlying distribution accurately.

Moreover, In Section 6, we conduct a broad ablation study to examine the contribution of different components of the proposed method. Regarding *Within-domain Tuning* (z-step EM), Fig. 16 shows a clear performance gap between models with and without this refinement, demonstrating its effectiveness. For *Knowledge-transfer Tuning*, Fig. 17 illustrates that adding PE can degrade performance as the number of shots increases, since additional shots introduce more noisy PE information into the learning process. For the number of Gaussian components, Fig. 18 shows that increasing the number of components consistently improves performance, indicating that the added expressiveness helps capture more complex ECP patterns. Regarding Gaussian mixture weights, Table 6 compares fixed weights, randomly selected fixed weights, and learnable weights. We observe no significant performance difference across these settings. As discussed in Section 6.4, this is likely because the additional flexibility of learnable weights introduces instability, as a small perturbations in the weights can result in large changes in the means and covariances of the Gaussian components, offsetting any potential gain. Finally, we tested covariance matrices with a increasing expressiveness by varying the rank in Eq. (14). As shown in Table 7, higher-rank covariances do not outperform the diagonal baseline. As discussed in Section 6.5, when the number of learnable parameters becomes large relative to the available observations, both *Knowledge-transfer Tuning* and *Within-domain Tuning* become less robust. In contrast, increasing the number of Gaussian components does not harm stability, because each component remains an isotropic Gaussian that can be reliably learned even under few-shot conditions. Thus, we think increasing rank is a less stable approach than increasing the number of mixture components, given the current data scale.

Finally, in Section 5.6, we present an interpretability analysis of the proposed method. In Section 5.4, we compare the proposed method with several conditional generative model benchmarks and demonstrate its advantages in few-shot ECP generation by leveraging Transformer's modeling ability and removing PE. In Section 5.5, we further compare our method with fine-tuning baselines to show its efficiency in terms of required time. As mentioned in Section 1, data scarcity is widespread in energy systems due to metering errors and privacy restrictions. The proposed method is therefore particularly beneficial for consumption and generation profile modeling under limited data, and can support a range of higher-level tasks. For example, the generated data can be used to train downstream deep learning models or to support system-level analyzes. Moreover, given the rapid deployment of IoT-based measurement infrastructures, our method has the potential to be applied to other domains with similar data scarcity challenges, such as water consumption, heat demand, or smart-building sensor data.

However, we note that although the current method sufficiently meets the experimental requirements of the study, it remains constrained by certain assumptions, such as Gaussianity and the use of fixed covariance structures. These assumptions may limit its performance in domains that demand higher model expressiveness. Looking forward, we believe that two challenges should be addressed simultaneously to further improve few-shot ECP modeling, (1) the development of a more advanced domain knowledge transfer framework, potentially in the form of a foundation model, and (2) the use of more expressive distribution modeling techniques, such as copula-based representations, to better capture complex statistical dependencies.

8. Conclusion

In this paper, we proposed an FSL method for ECP modeling that combines a Transformer encoder with Gaussian Mixture Models. We evaluated the proposed method on five datasets: the 15-minute ECP dataset, 30-minute ECP dataset, 60-minute ECP dataset, 15-minute solar generation profile, and 15-minute transformer-level ECP dataset. The results show that our method outperforms benchmark models such as CVAE, CFLOW, CDDPM, and TimesNet in terms of distributional metrics (MMD, KL, KS, WD, MSE.A), and can accurately estimate the original ECP distribution using only a small portion of the dataset (e.g., as little as 1.6% of the complete domain dataset).

Furthermore, we compared our method with a general DL-based generative model and demonstrated that our proposed approach is *order-invariant*, which contributes to its superior performance compared to CVAE. We also showed that our method is significantly more efficient than the classical fine-tuning mechanism. By replacing gradient descent-based model training/tuning for the target domain with EM in batch, our method achieved a speedup of thousands to potentially more than *thousands of times*. As a result, the proposed method is highly scalable and well-suited for applications involving thousands or even millions of domains.

We also provided interpretability and ablation studies to analyze the behavior of the framework. These results demonstrate how individual ECP samples influence GMM parameter estimation, and show that the effectiveness of *Within-domain Tuning*, removal of PE, and a lightweight GMM structure forms a stable and effective design. Increasing the number of Gaussian components improves expressiveness, whereas increasing covariance rank offers no benefit under few-shot conditions.

Looking ahead, the framework's Gaussian assumptions may limit its expressiveness for highly skewed or heavy-tailed loads. Future work includes integrating more advanced distribution models such as copulas, and exploring foundation-model–level pretraining to enhance transferability under extreme data scarcity. Beyond energy systems, the method can also benefit other IoT domains with limited observations, such as water usage, heat demand, and smart-building sensing.

CRediT authorship contribution statement

Weijie Xia: Writing – original draft, Visualization, Methodology, Investigation, Formal analysis, Conceptualization. **Gao Peng:** Writing – review & editing, Conceptualization. **Chenguang Wang:** Writing – review & editing, Conceptualization. **Peter Palensky:** Project administration, Funding acquisition. **Eric Pauwels:** Writing – review & editing. **Pedro P. Vergara:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This publication is part of the project ALIGN4Energy (with project number NWA.1389.20.251) of the research program NWA ORC 2020 which is (partly) financed by the Dutch Research Council (NWO), The Netherlands. This research utilized the Dutch National e-Infrastructure with support from the SURF Cooperative, The Netherlands (grant number: EINF-12580).

Appendix

Multi-Kernel Maximum Mean Discrepancy . The MMD, or specifically, Multi-Kernel MMD (MK-MMD), is defined as

$$\begin{aligned}\text{MK-MMD}^2(\mathbb{O},\mathbb{G}) = &\frac{1}{N(N-1)}\sum_{i\neq j}^{N}\sum_{p=1}^{P}k_p(x_i,x_j)\\ &-\frac{2}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{p=1}^{P}k_p(x_i,x'_j)\\ &+\frac{1}{N(N-1)}\sum_{i\neq j}^{N}\sum_{p=1}^{P}k_p(x'_i,x'_j),\end{aligned} \tag{15}$$

where k_p denotes the pth kernel function (Gaussian kernel), $x_i \sim \mathbb{O}$ are ECP samples drawn from the original dataset (target domain), and $x'_j \sim \mathbb{G}$ are ECP samples generated from the modeled GMMs. We use an equal number of samples from both distributions, denoted as N. All kernels are equally weighted in this formulation.

Kullback–Leibler Divergence (KL). For two probability densities $p(\mathbf{x})$ and $q(\mathbf{x})$, the KL divergence is defined as

$$D_{\text{KL}}(p \parallel q) = \int p(\mathbf{x}) \log\left(\frac{p(\mathbf{x})}{q(\mathbf{x})}\right) d\mathbf{x}. \tag{16}$$

In practice, we approximate the integral empirically over samples from p and q.

Wasserstein Distance (WD). Given distributions P and Q, the 1-Wasserstein distance is

$$W(P,Q) = \inf_{\pi\in\Pi(P,Q)}\int_{\mathbb{R}^d\times\mathbb{R}^d}\|\mathbf{x}-\mathbf{y}\|\, d\pi(\mathbf{x},\mathbf{y}), \tag{17}$$

where $\Pi(P,Q)$ denotes all joint distributions with marginals P and Q. In experiments, the distance is computed using empirical samples from each distribution.

Kolmogorov–Smirnov Distance (KS). For empirical CDFs $F_P(\mathbf{x})$ of real samples and $F_Q(\mathbf{x})$ of generated samples,

$$D_{\text{KS}}(P,Q) = \sup_{\mathbf{x}}\left|F_P(\mathbf{x}) - F_Q(\mathbf{x})\right|. \tag{18}$$

MSE of Autocorrelation (MSE.A). Let $R(\mathbb{O})$ and $R(\mathbb{G})$ denote the empirical autocorrelation vectors computed from the original and generated datasets, respectively. The metric is defined as

$$\text{MSE.A} = \frac{1}{T}\sum_{t=1}^{T}\left(R_t(\mathbb{O}) - R_t(\mathbb{G})\right)^2, \tag{19}$$

where T is the maximum lag considered. A smaller value indicates that the temporal correlation structure of the generated data better matches the real data.

Data availability

The link to the data is shared in the paper.

References

[1] Sharma A, Jain SK. A novel two-stage framework for mid-term electric load forecasting. IEEE Trans Ind Inform 2023;20(1):247–55.
[2] Tu C, Liu J, Wang J, Bai J, Hu G. High-resolution simulation and prediction of urban private vehicles energy consumption system: Agent-based modelling. Int J Electr Power Energy Syst 2025;168:110669.
[3] Duque EMS, Giraldo JS, Vergara PP, Nguyen PH, Van Der Molen A, Slootweg J. Risk-Aware Operating Regions for PV-rich distribution networks considering irradiance variability. IEEE Trans Sustain Energy 2023;14(4):2092–108.
[4] Sun M, He L, Zhang J. Deep learning-based probabilistic anomaly detection for solar forecasting under cyberattacks. Int J Electr Power Energy Syst 2022;137:107752.
[5] Kamana-Williams BLM, Gnoth D, Hooper R, Chase JG. A generalisable agent-based model of residential electricity demand for load forecasting and demand response management. Int J Electr Power Energy Syst 2025;168:110671.

[6] Lei M, Mohammadi M. Hybrid machine learning based energy policy and management in the renewable-based microgrids considering hybrid electric vehicle charging demand. Int J Electr Power Energy Syst 2021;128:106702.

[7] Xia W, Wang C, Palensky P, Vergara PP. A flow-based model for conditional and probabilistic electricity consumption profile generation. Energy AI 2025;100586.

[8] Xia W, Huang H, Duque EMS, Hou S, Palensky P, Vergara PP. Comparative assessment of generative models for transformer-and consumer-level load profiles generation. Sustain Energy Grids Netw 2024;38:101338.

[9] Rizzato M, Morizet N, Maréchal W, Geissler C. Stress testing electrical grids: Generative adversarial networks for load scenario generation. Energy AI 2022;9:100177.

[10] Lin N, Palensky P, Vergara PP. Energydiff: Universal time-series energy data generation using diffusion models. IEEE Trans Smart Grid 2025.

[11] Grigoras G, Cartina G, Bobric E, Barbulescu C. Missing data treatment of the load profiles in distribution networks. In: 2009 IEEE bucharest powerTech. IEEE; 2009, p. 1–5.

[12] Ding Y, Wang B, Wang Y, Zhang K, Wang H. Secure metering data aggregation with batch verification in industrial smart grid. IEEE Trans Ind Inform 2020;16(10):6607–16.

[13] Smart Grids Task Force EG. My energy data: A country-by-country overview of smart meter data availability in Europe. Tech. rep., European Commission, Directorate-General for Energy; 2016, Prepared by an ad hoc group of Expert Group 1 (Standards & Interoperability), European Smart Grids Task Force. URL https://energy.ec.europa.eu/system/files/2016-11/report_final_eg1_my_energy_data_15_november_2016_0.pdf.

[14] Azad S, Ameli MT. A domain adaptation-based convolutional neural network incorporating data augmentation for power system dynamic security assessment. J Eng 2024;2024(7):e12400.

[15] Yang S, Liu L, Xu M. Free lunch for few-shot learning: Distribution calibration. In: International conference on learning representations. 2021.

[16] Kong Z, Goel A, Badlani R, Ping W, Valle R, Catanzaro B. Audio flamingo: A novel audio language model with few-shot learning and dialogue abilities. In: Forty-first international conference on machine learning. 2024.

[17] Ojha U, Li Y, Lu J, Efros AA, Lee YJ, Shechtman E, Zhang R. Few-shot image generation via cross-domain correspondence. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021, p. 10743–52.

[18] Bishop CM. Mixture Density Networks. Aston University; 1994.

[19] Makansi O, Ilg E, Cicek O, Brox T. Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 7144–53.

[20] Bertinetto L, Henriques JF, Valmadre J, Torr P, Vedaldi A. Learning feed-forward one-shot learners. Adv Neural Inf Process Syst 2016;29.

[21] Ma Y, Zhong G, Wang Y, Liu W. Metacgan: A novel gan model for generating high quality and diversity images with few training data. In: 2020 international joint conference on neural networks. IEEE; 2020, p. 1–7.

[22] Kenton JDM-WC, Toutanova LK. Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of naacL-HLT, vol. 1, Minneapolis, Minnesota; 2019, p. 2.

[23] Song Y, Wang T, Cai P, Mondal SK, Sahoo JP. A comprehensive survey of few-shot learning: Evolution, applications, challenges, and opportunities. ACM Comput Surv 2023;55(13s):1–40.

[24] Mahdavi M, Javadi M, Wang F, Catalão JP. An accurate evaluation of consumption pattern in reconfiguration of electrical energy distribution systems. In: 2021 IEEE industry applications society annual meeting. IEEE; 2021, p. 1–7.

[25] Mahdavi M, Javadi MS, Wang F, Catalão JP. Optimal modeling of load variations in distribution system reconfiguration. In: 2021 IEEE international conference on environment and electrical engineering and 2021 IEEE industrial and commercial power systems Europe. IEEE; 2021, p. 1–6.

[26] Mahdavi M, Schmitt K, Chamana M, Bayne S. Distribution systems reconfiguration considering dependency of loads on grid voltage and temperature. IEEE Trans Power Deliv 2023;39(2):882–97.

[27] Mahdavi M, Awaafo A, Schmitt K, Chamana M, Jurado F, Bayne S. An effective formulation for minimizing distribution network costs through distributed generation allocation in systems with variable loads. IEEE Trans Ind Appl 2024;60(4):5671–80.

[28] Guo Z, Wang ZJ, Kashani A. Home appliance load modeling from aggregated smart meter data. IEEE Trans Power Syst 2014;30(1):254–62.

[29] Wagner C, Waniek C, Häger U. Modeling of household electricity load profiles for distribution grid planning and operation. In: 2016 IEEE international conference on power system technology. IEEE; 2016, p. 1–6.

[30] Singh R, Pal BC, Jabr RA. Statistical representation of distribution system loads using Gaussian mixture model. IEEE Trans Power Syst 2009;25(1):29–37.

[31] Duque EMS, Vergara PP, Nguyen PH, van der Molen A, Slootweg JG. Conditional multivariate elliptical copulas to model residential load profiles from smart meter data. IEEE Trans Smart Grid 2021;12(5):4280–94.

[32] Pillai GG, Putrus GA, Pearsall NM. Generation of synthetic benchmark electrical load profiles using publicly available load and weather data. Int J Electr Power Energy Syst 2014;61:1–10.

[33] Wang Z, Zhang H. Customized load profiles synthesis for electricity customers based on conditional diffusion models. IEEE Trans Smart Grid 2024;15(4):4259–70.

[34] El Kababji S, Srikantha P. A data-driven approach for generating synthetic load patterns and usage habits. IEEE Trans Smart Grid 2020;11(6):4984–95.

[35] Razghandi M, Zhou H, Erol-Kantarci M, Turgut D. Smart home energy management: VAE-GAN synthetic dataset generator and Q-learning. IEEE Trans Smart Grid 2023;15(2):1562–73.

[36] Vreeken J, Van Leeuwen M, Siebes A. Preserving privacy through data generation. In: Seventh IEEE international conference on data mining. IEEE; 2007, p. 685–90.

[37] Liu F, Cheng Z, Chen H, Wei Y, Nie L, Kankanhalli M. Privacy-preserving synthetic data generation for recommendation systems. In: Proceedings of the 45th international ACM SIGIR conference on research and development in information retrieval. 2022, p. 1379–89.

[38] Harder F, Adamczewski K, Park M. Dp-merf: Differentially private mean embeddings with randomfeatures for practical privacy-preserving data generation. In: International conference on artificial intelligence and statistics. PMLR; 2021, p. 1819–27.

[39] Yale A, Dash S, Dutta R, Guyon I, Pavao A, Bennett KP. Generation and evaluation of privacy preserving synthetic health data. Neurocomputing 2020;416:244–55.

[40] Zhang G, Cui K, Wu R, Lu S, Tian Y. PNPDet: Efficient few-shot detection without forgetting via plug-and-play sub-networks. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. 2021, p. 3823–32.

[41] Ma T, Zhang A. Affinitynet: semi-supervised few-shot learning for disease type prediction. In: Proceedings of the AAAI conference on artificial intelligence, vol. 33, no. 01, 2019, p. 1069–76.

[42] Zheng Z, Feng X, Yu H, Gao M. Cooperative density-aware representation learning for few-shot visual recognition. Neurocomputing 2022;471:208–18.

[43] Wu H, Zheng Z, Lv L, Zhang C, Bardou D, Niu S, Yu G. Dara: distribution-aware representation alignment for semi-supervised domain adaptation in image classification. J Supercomput 2025;81(2):376.

[44] Zheng Z, Wu H, Lv L, Zhang C, Guo H, Niu S, Yu G. Multi-branch semantic alignment for few-shot image classification. Inform Sci 2025;122676.

[45] Zheng Z, Zhu Y, Wu H, Lv L, Niu S, Yu G. SGE: Semantic-guided generalization enhancement for few-shot learning. Knowl-Based Syst 2025;113761.

[46] Wu H, Zhao Y, Li J. Invariant and consistent: Unsupervised representation learning for few-shot visual recognition. Neurocomputing 2023;520:1–14.

[47] Zhao Y, Ding H, Huang H, Cheung N-M. A closer look at few-shot image generation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022, p. 9140–50.

[48] Chen Y, Liu Z, Xu H, Darrell T, Wang X. Meta-baseline: Exploring simple meta-learning for few-shot learning. In: Proceedings of the IEEE/CVF international conference on computer vision. 2021, p. 9062–71.

[49] Wang Y, Wu C, Herranz L, Van de Weijer J, Gonzalez-Garcia A, Raducanu B. Transferring gans: generating images from limited data. In: Proceedings of the European conference on computer vision. 2018, p. 218–34.

[50] Li Y, Zhang R, Lu JC, Shechtman E. Few-shot image generation with elastic weight consolidation. Adv Neural Inf Process Syst 2020;33:15885–96.

[51] Zhao Y, Chandrasegaran K, Abdollahzadeh M, Cheung N-MM. Few-shot image generation via adaptation-aware kernel modulation. Adv Neural Inf Process Syst 2022;35:19427–40.

[52] Meng Y, Michalski M, Huang J, Zhang Y, Abdelzaher T, Han J. Tuning language models as training data generators for augmentation-enhanced few-shot learning. In: International conference on machine learning. PMLR; 2023, p. 24457–77.

[53] Nashid N, Sintaha M, Mesbah A. Retrieval-based prompt selection for code-related few-shot learning. In: 2023 IEEE/ACM 45th international conference on software engineering. IEEE; 2023, p. 2450–62.

[54] Akyürek E, Damani M, Zweiger A, Qiu L, Guo H, Pari J, Kim Y, Andreas J. The surprising effectiveness of test-time training for few-shot learning. 2024, arXiv preprint arXiv:2411.07279.

[55] Bazzani L, Larochelle H, Torresani L. Recurrent mixture density network for spatiotemporal visual attention. 2016, arXiv preprint arXiv:1603.08199.

[56] Tosi F, Liao Y, Schmitt C, Geiger A. Smd-nets: Stereo mixture density networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021, p. 8942–52.

[57] Shirsat A, Tang W. Quantifying residential demand response potential using a mixture density recurrent neural network. Int J Electr Power Energy Syst 2021;130:106853.

[58] Bishop CM, Nasrabadi NM. In: Pattern recognition and machine learning, vol. 4, no. 4, Springer; 2006.

[59] Zhang H, Liu Y, Yan J, Han S, Li L, Long Q. Improved deep mixture density network for regional wind power probabilistic forecasting. IEEE Trans Power Syst 2020;35(4):2549–60.

[60] Errica F, Bacciu D, Micheli A. Graph mixture density networks. In: International conference on machine learning. PMLR; 2021, p. 3025–35.
[61] McLachlan GJ, Lee SX, Rathnayake SI. Finite mixture models. Annu Rev Stat Appl 2019;6(1):355–78.
[62] Wu H, Hu T, Liu Y, Zhou H, Wang J, Long M. TimesNet: Temporal 2D-variation modeling for general time series analysis. In: The eleventh international conference on learning representations.
[63] Goswami M, Szafer K, Choudhry A, Cai Y, Li S, Dubrawski A. MOMENT: A family of open time-series foundation models. In: Forty-first international conference on machine learning.
[64] Rasul K, Seward C, Schuster I, Vollgraf R. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In: International conference on machine learning. PMLR; 2021, p. 8857–68.
[65] Ho J, Jain A, Abbeel P. Denoising diffusion probabilistic models. Adv Neural Inf Process Syst 2020;33:6840–51.
[66] Chefer H, Gur S, Wolf L. Transformer interpretability beyond attention visualization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2021, p. 782–91.
[67] Zien A, Krämer N, Sonnenburg S, Rätsch G. The feature importance ranking measure. In: Joint European conference on machine learning and knowledge discovery in databases. Springer; 2009, p. 694–709.
[68] Brandsæter A, Glad IK. Shapley values for cluster importance: How clusters of the training data affect a prediction. Data Min Knowl Discov 2024;38(5):2633–64.
[69] Antwarg L, Miller RM, Shapira B, Rokach L. Explaining anomalies detected by autoencoders using Shapley Additive Explanations. Expert Syst Appl 2021;186:115736.
[70] Sundararajan M, Najmi A. The many Shapley values for model explanation. In: International conference on machine learning. PMLR; 2020, p. 9269–78.
[71] Lundberg SM, Lee S-I. A unified approach to interpreting model predictions. Adv Neural Inf Process Syst 2017;30.
[72] Song J, Meng C, Ermon S. Denoising diffusion implicit models. 2020, arXiv preprint arXiv:2010.02502.
[73] Zhao Y, Li J, Gong Y. Low-rank plus diagonal adaptation for deep neural networks. In: 2016 IEEE international conference on acoustics, speech and signal processing. IEEE; 2016, p. 5005–9.