Individualized Forecasting of Gas Consumption Guided by Smart Meter Data Through Transform Integrated Neural Network

Xudong Hu and Biplab Sikdar, *Senior Member, IEEE*

***Abstract*—The accurate, extended period prediction of individual customer energy consumption is critical for utility providers. Machine learning techniques, particularly neural networks, have proven effective in predicting household energy consumption by identifying correlations and patterns. However, these predictions often generalize across the entire dataset, neglecting the distinct behaviors of specific sub-groups. This paper presents an innovative transformation architecture aimed at enhancing the prediction of gas consumption for multiple households or population subgroups concurrently. The adaptability of the transformation layer to various neural network frameworks allows for broader applicability. The model's performance is assessed based on prediction accuracy and efficiency. Furthermore, as the transformation layer may introduce private information during training, we also evaluate the robustness of the model against inference attacks and its resilience to Additive White Gaussian Noise (AWGN) and adversarial examples. Our results demonstrate that the proposed approach not only achieves parallel prediction with high accuracy but also maintains the ability to forecast consumption over an extended period without the need for recent meter readings.**

***Index Terms*—AIoT, extended period prediction, gas consumption, machine learning, parallel processing, privacy attack, subgroup, transformation.**

I. Introduction

ENERGY management stands at the forefront of discussions on sustainability and economic efficiency. The significance of energy management is underscored by its potential to optimize costs and ensure the judicious use of resources. As the global community grapples with the challenges of climate change, there is a marked shift towards carbon neutralization and the adoption of green energy solutions. This transition is not only essential for environmental preservation but also offers avenues for economic growth and innovation. For instance, a study on green seaports [1] highlights the potential of integrating renewable energy sources into major infrastructural projects, leading to reduced emissions and enhanced operational efficiency. Furthermore, pricing optimization strategies tailored for green energy can incentivize its adoption and make it a more viable alternative for consumers and industries alike [2], [3]. As the world progressively moves towards a more sustainable future, the role of energy management in shaping this trajectory cannot be overstated.

Natural gas has emerged as a significant player in the global energy mix. According to the International Energy Agency (IEA), natural gas contributes to 23% of the world's electricity generation. This statistic highlights the growing reliance on natural gas, especially as countries shift towards cleaner energy alternatives to coal and oil. The rise in natural gas consumption over recent decades can be traced back to several factors, including technological advancements in extraction methods like hydraulic fracturing and the global drive towards low-carbon energy sources to combat climate change. As the emphasis on sustainable and eco-friendly energy solutions grows, the trajectory of natural gas consumption is expected to further rise in the foreseeable future.

Additionally, in many countries (e.g., Singapore and USA), piped natural gas is also commonly used as an energy source in residential complexes and individual houses. In both industrial and domestic settings, technological innovations are revolutionizing the way we perceive and manage energy consumption. A prime example of this is the smart meter. Traditionally, a smart meter was an electronic device that recorded consumption metrics like voltage and current [4]. Today, their purview has broadened to encompass utilities like water, heat, and gas. These advanced meters furnish detailed data that, when astutely analyzed, can bolster applications such as consumption forecasting and user behavior analytics. Such insights assist utilities in optimizing production and distribution, and help users in planning their consumption.

The pervasive adoption of smart meters has paved the way for a more sophisticated approach to gas management, enabling a dynamic equilibrium between supply and demand. In the complex domain of industrial cyber-physical systems (ICPS), precise energy consumption prediction for individual units is pivotal for enhancing system efficiency and management. Discrepancies between demand and supply can inflate costs and

Received 20 March 2024; revised 20 July 2024; accepted 25 August 2024. Date of publication 30 August 2024; date of current version 13 September 2024. This work was supported by the Singapore Ministry of Education Academic Research Fund Tier 1 under Grant R-263-000-D63-114. *(Corresponding author: Xudong Hu.)*

The authors are with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 117583 (e-mail: e0459193@u.nus.edu; bsikdar@nus.edu.sg).

induce price volatility. Consequently, precise forecasting of gas consumption is of paramount importance. Currently, machine learning methodologies are progressively being leveraged to augment system proficiencies in both residential [5], [6], [7], [8] and industrial [9], [10] settings. To enhance accuracy further, studies employing time-series data alongside recurrent neural networks [11], [12] offer a promising avenue for achieving refined short-term prediction resolutions. However, the discourse at extended period and larger-scale forecasting remains scant. This gap is primarily because extended period predictions necessitate data up to the forecasted period, which often remains inaccessible.

Yet, a significant portion of existing models predominantly centers on the collective behavior of broad population segments. Such models often risk unintentionally overlooking the subtle and distinct behaviours inherent to individual units or specific sub-groups, a critical aspect in the multifaceted environment of ICPS. Addressing these nuances individually would necessitate resource-intensive model construction and maintenance. Additionally, the advent of smart meters, while beneficial, also raises privacy concerns. Regulatory frameworks and policies have been enacted by many countries to guide data custodians, prompting them to adopt various learning techniques [3], [13], [14], [15] to mitigate the risk of disclosing private user information. Hence, privacy evaluations become paramount when sensitive information is integral to system development. This paper endeavors to address this open problem by emphasizing the prediction of individual or subgroup behaviors, while ensuring stringent privacy safeguards.

A. Contribution

Conventional gas consumption prediction methods often fall short in analyzing extended period prediction, and individual pattern behavior for multiple targeted parties within a given system. Addressing this, our proposed transformation-based machine learning system offers four key advantages:

- Multi-output prediction: The proposed transformation layer reduces the low-level feature sharing and boosts independent learning for sub-groups.
- Extended period prediction: The proposed system can make predictions over extended periods since the actual consumption data is used as training guide at output side instead of a feature in the input matrix.
- Privacy inference protection: The sharing of low-level features learnt from other sub-groups can safeguard the private information from inference attack.
- Robustness: The transformation layer provides additional protection against noise and adversarial examples.

The remainder of this paper is structured as follows: Section II delves into the relevant literature and related works. Our proposed framework is described in Section III. Section IV presents the experimental results, validating the efficacy of our approach. The paper culminates with concluding remarks in Section V.

II. Related Work

Historically, time series models have been predominantly used for natural gas consumption prediction. The advent of smart meters has enabled the collection of high-resolution data, shedding light on individual consumption patterns. This granular data, whether from domestic [5], [6], [7] or industrial [9], [10] sectors, has paved the way for machine learning (ML) techniques to enhance prediction accuracy and introduce novel functionalities.

For instance, a comparative study in [24] evaluated six ML-based methods, each considering different household properties in London, to model in-building energy consumption. The findings underscored the superior performance of non-parametric methods, emphasizing the importance of non-linearity in capturing core behaviors. To further enhance performance, [25] introduced a hybrid algorithm-based model that amalgamates the strengths of various individual models. While this model boasts structural independence and achieves low errors across multiple time scales, it is susceptible to overfitting, making it less generalizable across different datasets.

In contrast, a neural network-based model presented in [26] offers next-day gas consumption predictions. Despite its simplistic architecture, this model underscores the potential of neural networks in this domain, albeit with certain data preprocessing and optimization prerequisites. An advanced iteration of this neural network model [27] incorporated weather conditions, enabling accurate predictions even in the absence of continuous meter data.

A common limitation among the aforementioned models is their treatment of training datasets as monolithic entities, thereby overlooking individual variations. For instance, while the model in [27] integrates multiple functional layers to enhance prediction accuracy, the models in [11] and [12] adopted the recurrent cell after convolution layer to acquire better prediction accuracy. They primarily capture the aggregate behavior of all users in the dataset of gas consumption. The recurrent implementation is also hard to make prediction for a longer term objective. A new model introduced in [8] highlighted the feasibility of achieving comparable accuracy by focusing on individual sub-group properties rather than the entire dataset. However, this model did not fully harness the efficiency-boosting potential. This gap is addressed by our proposed model, which parallelizes predictions across all sub-groups.

When models aim to incorporate advanced features like parallel processing, they must grapple with resource constraints and escalating complexity. Conventional regression models [16], [17] leverage linear correlations between time steps and employ multiple output regressions to optimize resource usage. However, these models are hamstrung by their inability to capture non-linear correlations and handle sequences of zeros, such as during prolonged absences of homeowners. Recognizing the significance of non-linear correlations, some researchers have endeavored to enhance model efficiency by either constructing multiple models or implementing multi-task learning (MTL) in neural networks [28], [29]. Recent literature has documented the proliferation of MTL applications across domains like Natural Language Processing (NLP) [18], [19] and computer vision [23], [30]. These models capitalize on shared low-level features to bolster task performance. Nevertheless, when applied to parallel processing, these models require a more independent analysis to discern sub-group behaviors. Furthermore, as more information

TABLE I
SUMMARY OF RELATED WORKS

Feature	[5],[6],[7], [9], [10]	[16], [17]	[11], [12],	[18],[19],[20], [21],[22],[23],	Proposed
Non-linear features	✓	✗	✓		✓
Parallel processing	✗	✓	✓		✓
Task independence	✗	✓	✗		✓
Sub-group property	✗	✓	✗		✓
Scalable	✗	✓	✗		✓
Extended period prediction	✗	✓	✗		✓
Privacy assessment	✗	✗	✗		✓

– *Non-linear features:* Does the method capture non-linear features to perform the task?
– *Parallel processing:* Does the method perform multiple goals in parallel?
– *Task Independence:* Does the method protect performance from being affected by other tasks?
– *Sub-group property:* Does the method distinguish the behavior from individual sub-groups?
– *Scalable:* Does the method scale with the number of tasks?
– *Extended period prediction:* Does the method perform prediction for a later time?
– *Privacy assessment:* Does the method consider the privacy risk?

is incorporated to advance system objectives, sensitive data involved during training can become susceptible to privacy breaches. For instance, when tasks are processed together within the same network as in MTL, sharing information between tasks is inevitable; this makes sensitive information vulnerable to inference attacks. Assessing privacy risks is critical, as the pursuit of advanced system objectives should not compromise privacy.

Table I encapsulates the constraints of existing methods and underscores the advantages of our proposed approach.

III. METHODOLOGY

This section describes the proposed system, segmenting it into its core components: dataset, network architecture, and evaluation method. We further delve into privacy considerations and the system's resilience against challenges such as noise and adversarial attacks.

A. Dataset

Similar to the system proposed in [27], we integrate weather data, day of the week, and specific house-related metrics into our input matrix to forecast gas consumption. This matrix merges data from dwelling and weather sources. Our dwelling data originates from Pecan Street [31], spanning almost two years of gas meter readings (from September 2015 to July 2017) across 155 dwellings, inclusive of house-specific details. Concurrently, our weather data is sourced from Kaggle [32], providing daily weather records for the identical period.

Utilizing a dataset from real households enhances the practicality of our framework, establishing a realistic benchmark for subsequent industrial applications. Nevertheless, this dataset presents challenges, including inconsistencies, missing values, and erroneous entries. To synchronize the granularity of our weather and consumption data, we compute the daily consumption by subtracting the day's initial reading from its final one. We omit entries with absent meter readings to avert training biases. The approach for managing missing feature data is outlined in Table II. Besides, to facilitate predictions over extended periods, the actual consumption data is deliberately excluded from the

TABLE II
SELECTED FEATURES AND DESCRIPTION

Feature	Elements	Description
Temp	1	Average reading in the day
Humidity	1	Average reading in the day
Wind	1	Average reading in the day
Precipitation	1	Average reading in the day
Day in the week	7	Presented by one hot encoding
Year of construction	1	Years from completion to current year (The mean value of the dataset is selected for missing value replacement)
Footage	1	Total footage of the house (The mean value of the dataset is selected for missing value replacement)
Number of dryers	1	Total number of dryers using gas. (0 is selected for missing value replacement)
Number of furnaces	1	Total number of furnaces using gas (0 is selected for missing value replacement)
Number of heaters	1	Total number of heaters using gas (0 is selected for missing value replacement)

input matrix. This approach addresses the challenge of requiring data up to the forecasted period, which often remains unavailable. Instead, the actual consumption data serves as a guiding reference on the output side during the training phase, enabling the model to learn and predict future consumption patterns without the immediate need for up-to-date input data. This method ensures the model's capability to generate long-term forecasts while navigating the limitations posed by the unavailability of future data. In the final stage of data preprocessing, to rectify any anomalous entries, we removed records with missing or incorrect meter readings, thus ensuring a robust training and testing dataset. The dataset is then randomly partitioned, with 90% allocated for training and 10% for testing, to ensure thorough training across all residences and minimize the risk of overfitting. However, to simulate the data in a time-series format with the challenges mentioned earlier, we took additional steps to partition the dataset for the CNN-LSTM and CNN-BiLSTM models. These steps will be discussed in detail in Section IV-A for a comprehensive evaluation.

B. Network Architecture

In contrast to the large data volumes required for image representation, the total number of chosen features in our scenario amounts to only 16. To process multiple dwellings in parallel, vector stacking presents several drawbacks:

– Producing a non-square input if the number of groups doesn't match the total number of selected features.
– Compelling the neural network to prioritize inter-subgroup correlation if convolution operations are involved.
– Exposing sensitive information, such as house ID.

To optimize efficiency and safeguard private data within the data group, we introduce a novel system equipped with a transformation layer. An appropriate linear transformation can fulfill our objectives without altering the dataset's properties. Given

data x traversing a feed-forward hidden layer, the output y is expressed as:

$$y = \sigma(w \cdot x + b)$$

where σ is the activation function, w represents the weights, and b is the bias.

With the transformation, the output data can be derived as:

$$x' = Ax + c$$
$$y' = \sigma(w \cdot (Ax + c) + b)$$

where A is the transformation matrix and c is the bias vector.

Thus, we can achieve the objective with the new weights and bias as:

$$y' = \sigma(w' \cdot x + b')$$

where $w' = wA$ and $b' = w \cdot c + b$.

Besides, the transformation matrix should maximize the independence between subgroups. Similar to the use in wireless communications and signal processing, we use Fourier transform to convert a time domain signal to its frequency domain representation. Through transformation, multiple signals can be transmitted together while preserving their information and not interfere with each other. However, the output of Fourier transformation and its inverse transformation contains the operation of complex numbers. To achieve the real-valued transformation, two dimension Discrete Cosine Transform (2D-DCT) is adopted which is defined as:

$$X_{k1,k2} = \sum_{n1=0}^{N_1-1} \sum_{n2=0}^{N_2-1} x_{n1,n2} \cos\left[\frac{\pi}{N_1}\left(n_1 + \frac{1}{2}\right)k_1\right] \cdot \cos\left[\frac{\pi}{N_2}\left(n_2 + \frac{1}{2}\right)k_2\right] \quad k = 0, \ldots, N-1. \quad (1)$$

Here, $x_{n1,n2}$ are the values at position $n1$ and $n2$ and k_1 and k_2 are the indices of the DCT coefficients in the vertical and horizontal directions. 2D-DCT decomposes the message signal to a summation of different basis functions. This transformation is applied to both axes to obtain the two dimensional transformation. As a result, the message signals are transferred to a function summation with same matrix size that has the properties of linearity, real values, as well as independence to each other.

To protect the private information, house ID in our specific environment, we define a dictionary that will map the house ID to the position where the post transformation message is located. The arrangement of the matrix that contains all subgroup information is set to size of 52×52. This is calculated based on the boundary condition criteria defined in the equations below:

$$\min_L : \left(\frac{L}{l}\right)^2 \geq N, \quad (2)$$

$$row = \left\lfloor i \div \frac{L}{l} \right\rfloor \times l, \quad (3)$$

$$column = \left(i \mod \frac{L}{l}\right) \times l. \quad (4)$$

Algorithm 1: Message Signal Transformation and Mapping.

input : $Message\ signal\ s_i\ from\ one\ subgroup$
output: $Post-transformation\ signal\ S\ from\ the\ day$

$Init\ F\{S\}\ with\ 0s$

$//\ Transformation\ and\ mapping\ of\ single\ message$
$s_i \xrightarrow{2D-DCT} F\{s_i\}$
$Mod\_Dic[i] \leftarrow location\ index\ number$
$r_i,\ l_i \leftarrow row\ and\ col from\ Eqn.\ (3),\ (4)$
$F\{S\}[r_i, l_i] = F\{s_i\}$
$//\ end\ of\ single\ message\ signal$

if $multi-output$ **then**
  while $date == date\_to\_process$ **do**
    for $i\ in\ I_*,\ I_* \in \Theta\ \forall * = 1, 2, 3 \cdots N$ **do**
      $//\ \Theta = all\ possible\ house\ ids$
      $repeat\ transformation\ for\ I_*[i]$
    end
  end
end
$F\{S\} \xrightarrow{inverse\ 2D-DCT} S$

Here, L represents the number of mapping matrices, l denotes the length of the message signal, i indicates the location index number, and N is the total number of subgroups. Specifically, in our scenario, the data corresponds to 155 dwellings and thus, N=155. Also, the input message is composed of 16 elements, organized in a 4×4 matrix, which sets l at 4. Applying these values in (2), we determine L to be 52.

Algorithm 1 describes the detailed operations in the transformation layer. The computational complexity of the transformation layer depends on the total number of subgroups. A physical system only needs to define the total number of subgroups which contribute to $O(n)$ complexity where n is the number of subgroups. Fig. 1 illustrates the physical meaning of each step in the transformation layer. The input data from each user with 16 elements is mapped to a 4×4 block as an individual message for the subgroup. We create a dictionary that has a one-to-one mapping from subgroup data-id to a number that ranges from 0 to 154. The mapped number is the location index that defines where the message signal maps into the mapping matrix. The position is described with index of row in (3) and column in (4) to represent where the 4×4 matrix is placed. Step 1 to step 3 is repeated for each subgroup to complete the whole mapping matrix with all the subgroup messages. Then, the mapping matrix undergoes a process of inverse discrete cosine transform (IDCT) before being passed to the next layer. Since the transformed signal after mapping is sparse under the current set up, cross interference and aliasing is not discussed in this work.

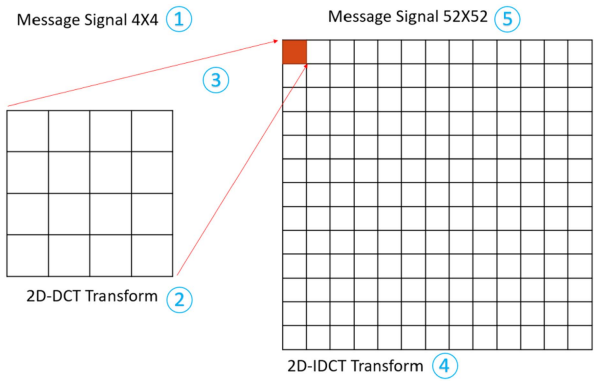

Fig. 1. Transformation and mapping of input signals. Step 1: construct the 4×4 matrix with 16 elements of subgroup input information; Step 2: apply 2D-DCT transformation on the 4×4 signal; Step 3: move the message to the 52×52 mapping matrix based on the location index; Step 4: apply inverse 2D-DCT to the 52×52 mapping matrix; Step 5: pass the 52×52 signal to the convolution layer.

Overall architecture of the proposed system is shown in Fig. 2. Convolutional layers are well known for their good performance in the feature capturing task. The middle layers consist of convolution layer, Rectified Linear Unit (LeakyRelu), max pooling layer and average pooling layer. Fully connected layer is attached to the end to perform the prediction task. To improve the prediction accuracy of the proposed system, we introduce another parameter as an input that intends to capture the presence of an occupant in the dwelling. The parameters acts as an indicator function that takes on a value of 1 if the dwelling is occupied at that point of time, and 0 otherwise. This presence indicator is generated from the input data based on the value in the consumption column and the indicator's value is 0 if the reported consumption is 0, and the value is 1 otherwise.

C. Loss Function and Optimizer

The loss function and optimizer symbolize the features learned from the data by updating the parameters via backpropagation in each iteration. In our proposed system, we adopted both Mean Absolute Error (MAE) and the mean squared error (MSE), which are defined as:

$$MAE\ Loss = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right| \tag{5}$$

and

$$MSE\ Loss = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2 \tag{6}$$

where $\hat{Y}_i$ is the predicted output from the model for the i-th day and Y_i is the actual consumption of that day.

The MAE loss is more robust to outliers since it only compares the absolute difference between the predicted value and actual value. On the other hand, although MSE loss overloads penalties on the extreme records with very high consumption, it is efficient in finding optimal points and faster to achieve convergence. During the parallel processing for multiple households, the total loss function can be formulated as

$$L_{Total} = \sum_{i}^{N} \omega_i \cdot L_i \tag{7}$$

where ω_i and L_i are the weight and loss associated with household i, respectively. Since all subgroups are designed to be independent, ω_i are the same for all subgroups. Adam optimizer [33] is chosen to update model parameters rather than Stochastic Gradient Descent as the optimizer to achieve a better performance during the training stage.

D. Feature Scaling

Features with varying units necessitate appropriate scaling when amalgamated to form the message signal. For instance, weather metrics like precipitation are substantially smaller than house footage in magnitude. A dominant feature can overshadow the influence of other features. To counteract this under the assumption that all selected features are equally impactful, we preprocess data before the transformation stage, fitting the features to a consistent scale using the weight vector λ_j. Additionally, signal amplification with factor A is employed during the transformation to enhance the signal level, thereby bolstering noise resistance. The overall transformed signal is:

$$F_j = A \times F\left\{\lambda_j * f\right\} \tag{8}$$

where F_i and f_i are the transformed and original feature j, respectively.

E. Evaluation

The prediction performance is gauged using the Mean Absolute Percentage Error (MAPE), MAE and MSE. MAPE is defined as:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{Y_i - \hat{Y}_i}{Y_i}\right|. \tag{9}$$

MAPE is involved in the evaluation since it better captures the accuracy performance as compared to absolute difference, similar to the comparison between MAE and MSE. When the consumption is high, a small percentage error still produces a higher error rate when absolute error calculation is used.

IV. Results

This section discusses the evaluation of our proposed prediction framework's performance. We present results from experiments conducted on both single-output and multi-output parallel models, which predict multiple households simultaneously. Both models were tested on an NVIDIA A5000 graphics card.

A. Performance

As discussed in Section III-A, outliers in the dataset can originate from a variety of sources such as human behavior or sensor malfunctions. An example includes instances where a residence's raw data might suggest an unrealistic consumption

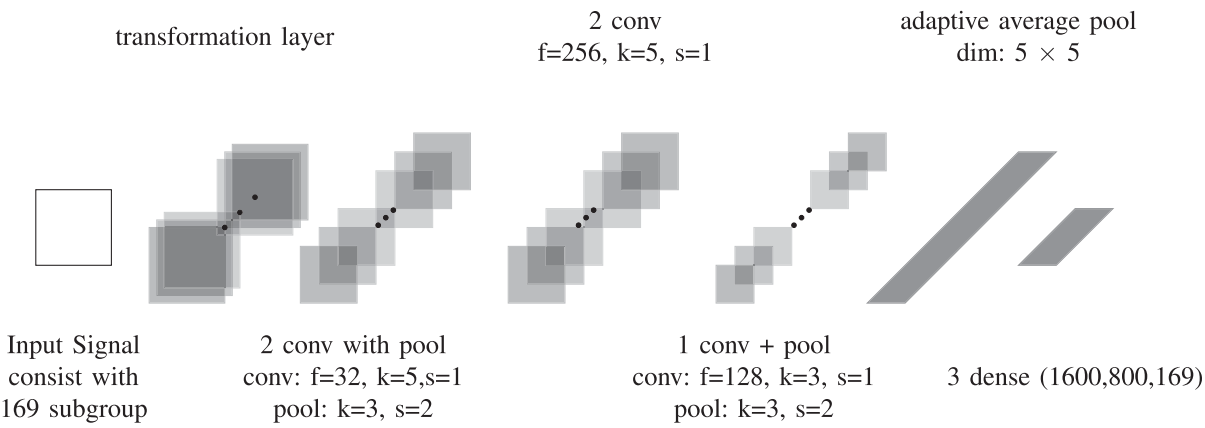

Fig. 2. The architecture of the neural network in proposed system. Conv stands for convolution layer; pool stands for maxpooling layer; f stands for number of filters; k stands for kernel size; s stands for stride.

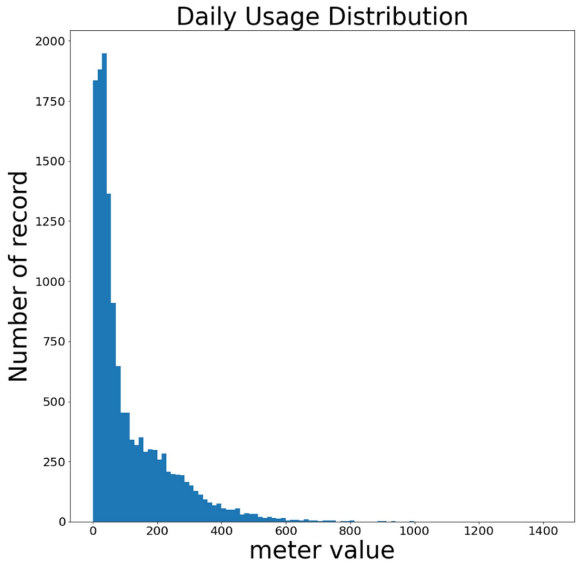

Fig. 3. Raw daily gas consumption record for one dwelling.

TABLE III
TIME-AVERAGE PERFORMANCE SUMMARY

	MAPE	MAE	RMSE
7-day average	1.17	41.20	70.36
30-day average	1.32	48.29	78.76

exceeding 1000 cubic feet of gas, as illustrated in Fig. 3. Removing these outliers is essential for stabilizing model performance and hastening convergence.

Given the simplicity and efficiency of time-average prediction, we conducted an experiment to evaluate the necessity of a complex model for each household's prediction task. Table III summarizes the average performance of both 7-day and 30-day average prediction methods for all households. Fig. 4 depicts the discrepancy between actual consumption and the predicted values for both methods. We observed significant differences

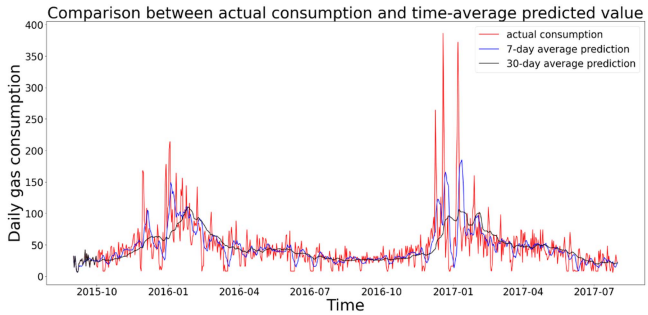

Fig. 4. Comparison between actual consumption and time-averaged consumption prediction.

during periods of rapid consumption increase, where the prediction methods failed to keep pace, highlighting the need for a more sophisticated model to enhance prediction accuracy.

Table IV compares our method and its variations with the CNN model from [27], as well as CNN-LSTM [11] and CNN-BILSTM [12] models. The results reveal that our proposed transformation layer approach, even when combined with other data augmentation methods like the Gramian Angular Field (GAF) image representation, effectively adapts to various network architectures for parallel processing tasks. As explored in Section II, several strategies exist for parallel processing. Multi-Task Learning (MTL), for instance, relies on shared low-level features, but different subgroups may interfere with each other's learning. Our transformation approach aims to keep signals as independent as possible, reducing the influence on adjacent signals. Another strategy involves populating data directly in the subgroup space without transformation, which, although simpler, sacrifices accuracy (as evidenced in Table IV) and increases privacy risks.

As mentioned in Section II, LSTM and BILSTM are leveraged for time-series data processing to enhance accuracy. We constructed CNN-LSTM and CNN-BILSTM models (details in Table V) for a comparative performance analysis. Inspired by [11], we incorporated meter readings as an additional feature,

TABLE IV
COMPARISON OF MODEL PERFORMANCE ACROSS DIFFERENT IMPLEMENTATIONS

Architecture	Transformation	Data augmentation	Configuration	Loss Function	MAPE	MAE	RMSE
Convolution	No	N.A.	Single-output	MSE	0.26	9.17	14.98
				MAE	0.25	8.77	14.33
	Yes		Single-output	MSE	0.23	8.18	13.36
				MAE	0.18	6.18	10.42
	Yes		Multi-output	MSE	0.24	8.37	13.68
				MAE	0.16	5.58	9.12
	No	Apply image representation	Single-output	MSE	0.46	16.15	26.39
				MAE	0.45	15.75	25.73
	Yes		Multi-output	MSE	0.40	14.16	23.13
				MAE	0.33	11.56	18.89
Convolution+LSTM	N.A.	sampled by time-end method	Single-output	MSE	0.36	12.79	20.90
				MAE	0.40	14.22	23.23
		sampled by house-id method		MSE	0.59	20.75	33.90
				MAE	0.54	19.01	31.06
		sampled by random method		MSE	0.46	16.22	26.50
				MAE	0.49	17.14	28.00
	DCT	sampled by time-end method	Multi-output	MSE	0.60	21.21	34.65
				MAE	0.56	19.81	32.36
		sampled by house-id method		MSE	0.65	23.08	37.71
				MAE	0.62	21.95	35.86
		sampled by random method		MSE	0.57	20.26	33.10
				MAE	0.58	20.35	33.25
Convolution+ Group_LSTM	DCT	sampled by time-end method	Multi-output	MSE	0.45	15.97	26.09
				MAE	0.39	13.78	22.51
		sampled by house-id method		MSE	0.58	20.44	33.39
				MAE	0.58	20.45	33.41
		sampled by random method		MSE	0.55	19.47	31.81
				MAE	0.53	18.68	30.52
Convolution+BILSTM	N.A.	sampled by time-end method	Single-output	MSE	0.21	7.25	11.84
				MAE	0.17	5.87	9.59
		sampled by house-id method		MSE	0.68	23.85	38.96
				MAE	0.58	20.43	33.38
		sampled by random method		MSE	0.48	16.90	27.61
				MAE	0.47	16.44	26.86
	DCT	sampled by time-end method	Multi-output	MSE	0.54	19.16	31.30
				MAE	0.54	19.03	31.09
		sampled by house-id method		MSE	0.73	25.91	42.33
				MAE	0.68	23.97	39.16
		sampled by random method		MSE	0.67	23.49	38.38
				MAE	0.60	21.01	34.32
Convolution+ Group_BILSTM	DCT	sampled by time-end method	Multi-output	MSE	0.37	12.88	21.04
				MAE	0.33	11.54	18.85
		sampled by house-id method		MSE	0.62	21.71	35.47
				MAE	0.61	21.67	35.40
		sampled by random method		MSE	0.54	18.96	30.98
				MAE	0.55	19.53	31.91

TABLE V
ARCHITECTURE OF CNN-LSTM AND CNN-BILSTM MODEL WITHOUT TRANSFORMATION LAYER

Layer	Filter	Kernel size	Stride
Convolution	32	(2,1)	(1,1)
batchnorm	32	-	-
leakyrelu	-	-	-
Convolution	64	(2,1)	(1,1)
batchnorm	64	-	-
leakyrelu	-	-	-
pooling	-	(2,1)	(2,1)
Convolution	256	(2,1)	(1,1)
batchnorm	256	-	-
leakyrelu	-	-	-
pooling	-	(2,1)	(2,1)
LSTM(128)[1]	-	-	-
attention(128)[1]	-	-	-
dense (64)	-	-	-
dense (1)	-	-	-

using a 30-day interval to structure the dataset in a 30×17 format via a sliding window algorithm. For models incorporating the transformation layer, we categorized households into four groups based on their average usage from low to high and arranged them sequentially on the time axis, as the network processes data in a single dimension in the CNN layer. Consequently, the input data for these models is resized to 120×17. Since the dataset is not well-structured in a time-series format, we employed three distinct methods to split the dataset for training and testing, ensuring a comprehensive evaluation:

- Time-end method: Sampling the final 10% of the time stamps from each household for testing.
- House-id method: Selecting 10% of households and using all their data as testing samples.
- Random: Randomly selecting 10% of all data as testing samples.

TABLE VI
ARCHITECTURE OF CNN-LSTM AND CNN-BILSTM MODEL WITH TRANSFORMATION LAYER

Layer	Filter	Kernel size	Stride
Convolution	32	(11,1)	(1,1)
batchnorm	32	-	-
leakyrelu	-	-	-
Convolution	64	(9,1)	(1,1)
batchnorm	64	-	-
leakyrelu	-	-	-
Convolution	256	(2,1)	(1,1)
batchnorm	256	-	-
leakyrelu	-	-	-
pooling	-	(2,1)	(2,1)
Convolution	256	(2,1)	(1,1)
batchnorm	256	-	-
leakyrelu	-	-	-
pooling	-	(2,1)	(2,1)
LSTM(512)[1]*	-	-	-
attention(512)[1]*	-	-	-
dense (64)	-	-	-
dense (4)	-	-	-

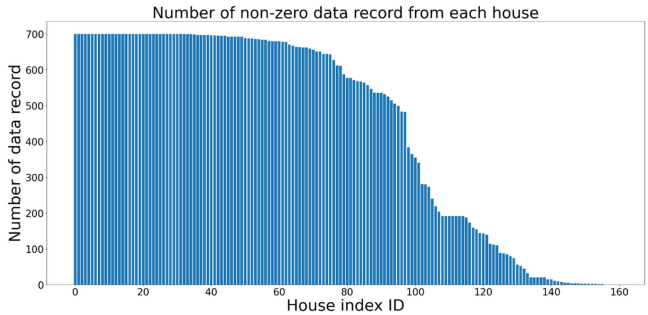

Fig. 5. Distribution of data record per house id before window sliding algorithm.

To illustrate that the proposed transformation layer is an innovative approach rather than a mere data augmentation method, we devised two architectures (as detailed in Table VI) for the CNN-LSTM and CNN-BILSTM models:

- Single recurrent cell: Utilizes only one set of recurrent cells but with four times the number of hidden units.
- Per group recurrent cell: Employs one recurrent cell per group, maintaining the same number of hidden units.

The results did not surpass the accuracy of the CNN model since the dataset does not strictly adhere to time-series data conventions, characterized by numerous missing values in timestamps, as depicted in Fig. 5. The absence of consistent timestamps disrupts temporal correlations and confounds the network, leading to comparable performances between LSTM and BILSTM implementations. We also found that the time-end method outperforms others as it best retains the time-series characteristics. Regarding the transformation layer, implementations using per group recurrent cells outperformed those with a single recurrent cell and were competitive with single-output implementations. Thus, we conclude that a strategic organization of the architecture with the transformation layer enables parallel processing capabilities in the model.

[1]* BILSTM will set bidirection=True and thus the hidden units will double.
[2]Per group recurrent implementation has one recurrent cell per group and thus lead to 4 group of 128 hidden units.

TABLE VII
SPEED COMPARISON BETWEEN SINGLE-OUTPUT AND PARALLEL SYSTEMS FOR NEURAL NETWORK PREDICTION

System	Speed(s)
Single-output (without recovery)	10.53
Single-output (with recovery)	92.96
Parallel (without recovery)	0.27
Parallel (with recovery)	1.19

Table VII compares processing times across different configurations. Without the recovery layer, our method achieves a 39-fold speedup when processing 155 dwellings concurrently. This acceleration factor can reach up to 78 when the recovery layer is incorporated. These results confirm that the proposed approach can significantly enhance prediction efficiency, with a minor 3% accuracy trade-off compared to a single-output model [8].

Fig. 6 shows the MAPE performance for each dwelling for both the single-output and multi-output parallel systems. Excluding outliers from the dataset significantly improves individual predictions. This improvement is attributed to the fact that, compared to the entire dataset, subgroups have fewer data records, making them more sensitive to outliers.

B. White-Box Location Index Inference Attack

In this section, we address potential privacy risks associated with our system. While the transformation layer facilitates parallel processing capabilities and allows the system to tailor learning based on distinct subgroup behaviors, it incorporates a location index to signify group identity, ensuring the model retains this information throughout its processing. This design aims to ensure that enhancements in system performance do not compromise privacy integrity.

However, when attackers possess knowledge of the network architecture and have access to a dataset, they could potentially pinpoint the exact household from which the data originated by accurately inferring the location index. This scenario is akin to a Membership Inference Attack [34], where information utilized during training retains its distinctiveness as a system bias. Notably, the location index, introduced as a novel attribute, maintains a direct link to individual dwelling identities. This unique identifier aids the system in recognizing the subgroup, thereby facilitating predictions based on collective subgroup behaviors. Consequently, when input data is accurately positioned, the system is expected to exhibit the lowest error rate. The primary attack strategy involves identifying the location index that corresponds to the lowest error rate across all possibilities, as outlined in Table VIII:

$$\hat{I} = \text{argmin}_{I \in \Theta} f\{X_I\} \tag{10}$$

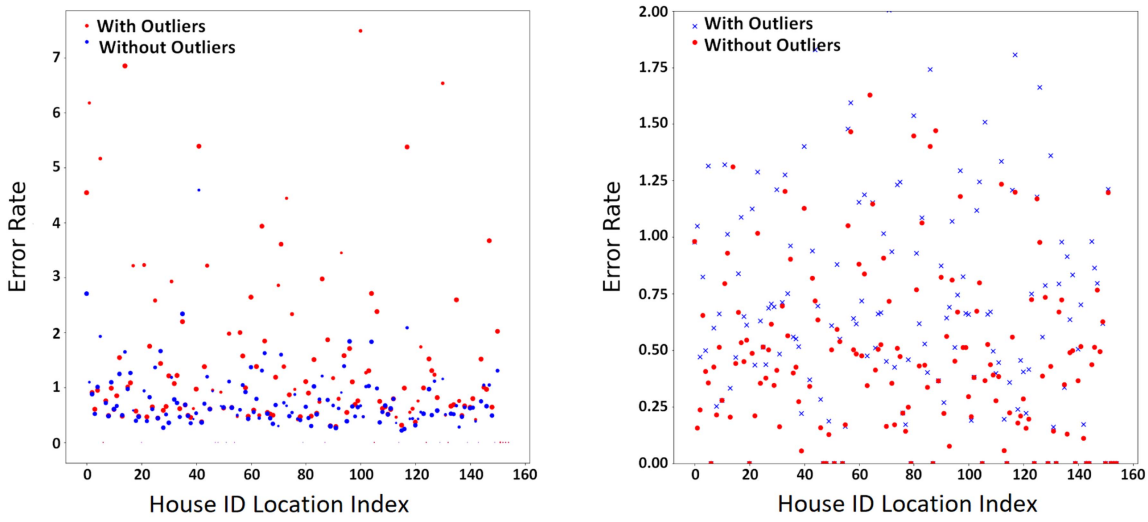

Fig. 6. MAPE performance for individual dwellings, with marker size indicating data population.

TABLE VIII
ROBUSTNESS AND PRIVACY EVALUATION SCENARIOS: DESCRIPTION AND ASSUMPTION

Scenario	Description	Assumption
White-box location index inference attack	• Attacker aims to find the index number of the dwelling house with a set of actual consumption data. • Attacker will exhausted the search for the goal. • System operate in ideal environment with Gaussian Noise	• Attacker has full knowledge about the network architecture. • Attacker use minimum error to decide the index number.
Additive white Gaussian noise(AWGN)	• Noise power is set at -20dB where the message signal power is around -9dB • System operate together with reconstruction layer inspired by Han and Qinkai [35]	• Communication channel is trusted and attack free
Adversarial Attack	• Attacker aims to poison the input such the system will not able to provide correct prediction • Attacker utilize Fast Gradient Sign Method to achieve minimum change in the input data	• Communication channel is noise free and trusted • Attacker does not have knowledge about the Network • Attacker need to avoid system abnormal detection at input

where Θ is the collection of all possible location indices and $f\{X_I\}$ represents the error value when the index is I. The attack is summarized in Algorithm 2.

Executing the attack as described in Algorithm 2 reveals that the attacker successfully acquires the correct location index in 45 out of 10,000 instances, indicating a 99% defense success rate for the system Fig. 7. To further elucidate these findings, we examined three specific attack instances, tracking the error rates associated with each location index per round. The actual location indices for these sampled attacks were 31, 68, and 146, respectively, marked with a red star in our analysis. The graphical representation of error rates correspond to the guessed location indexes during these attacks, and as shown in Fig. 8, indicate that the index correlating with the lowest error did not align with the actual location index. This diminished effectiveness of the attack method can primarily be attributed to the multi-output system's ability to learn not just individual behaviors but also inter-subgroup dynamics concurrently. When the system is provided with information from only one subgroup, its response diverges from the biased imprint left during training, rendering the multi-output method more robust against this category of inference attacks.

C. Robustness Against Noise and Adversarial Examples

Our model's robustness is evaluated from two perspectives, as detailed in Table VIII. Firstly, we consider the inevitable noise present in real-world systems. We assume that our system operates in a communication channel that, while noisy, is free from malicious attacks. Secondly, we address adversarial samples, which can be particularly detrimental to regression models. To enhance robustness against such threats, we introduce an interlayer, inspired by [35], to mitigate the impact of adversarial examples. This interlayer involves a bit-drop step (to minimize the influence of compromised bits) followed by a reconstruction layer that amplifies major signals in the mapping matrix while suppressing others.

We assume that attackers use the Fast Gradient Sign Method (FGSM) to subtly alter the input, thereby misleading the neural network without triggering abnormal system detection. The

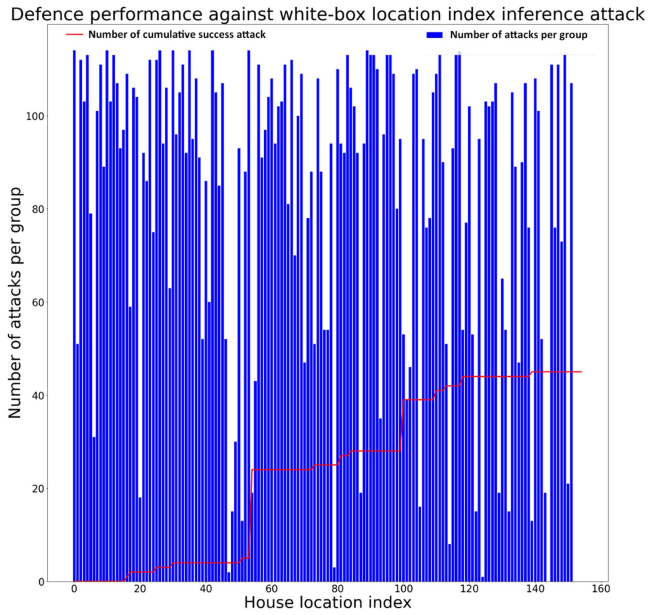

Fig. 7. Defence rate against white-box location index inference attack.

Algorithm 2: White-Box Location Index Inference Attack.

input : *Message signal* s_i *from one subgroup*
output: *Predicted location Index* I
for $I_* \in \Theta,\ \Theta =$ *All House_ID* **do**
  for i *in* $Range(0, N_s),\ N_s =$ *No. of subgroups* **do**
    $MAPE[i] = 0$
    $MAPE_{TMP} = 0$
    // *Get average MAPE of the data*
    for *date in range*$(0, N_d), N_d =$ *No. of input* **do**
      Init $F\{S\}$ *with* 0*s*
      Get $F\{S\}$ *from Algo.* 1
      $F\{S\} \xrightarrow{inverse\ 2D-DCT} S$
      Get $MAPE_d$
      $MAPE_{TMP} += MAPE_d$
    end
    $MAPE[i] = \frac{1}{N_d} * MAPE[i]$
  end
  $I[*] = arg\ \min_i MAPE[i]$
end

Signal to Noise Ratio (SNR) quantifies the noise level in the signal. The signal's power is computed as:

$$P_x = \frac{1}{N} \sum_{N-1}^{n=0} |x[n]|^2 \tag{11}$$

where N is the total number of samples in the signal, and $x[n]$ signifies the nth signal in the sequence

TABLE IX
MAPE Performance Comparison Between Single-Output and Multi-Output Models Under 0 dB Gaussian Noise and FGSM Attack

Condition		Single Output	Multi-output with transformation	
			Yes	No
Clean Channel	N.A.	33.5	36.3	67.5
With AWGN	with recovery	34.5	37.2	40.4
	without recovery	33.1	37.5	50.7
FGSM Attack	with recovery	60.3	43.0	N.A.
	without recovery	66.2	45.7	N.A.

This power can be converted to the dBW unit for easier interpretation:

$$P_{dBW} = 10 log P_x. \tag{12}$$

Given that the minimum signal power from our test dataset is approximately -9 dB, we set the noise power to -20 dB to simulate a 11 dB SNR scenario. As shown in Table IX, the multi-output model demonstrates resilience against noise. However, its performance declines under adversarial attacks. Despite this, the multi-output model outperforms the single-output model when faced with FGSM attacks.

Adversarial samples are generated by adding a gradient to the input, leading the neural network astray. The adversarial sample can be formulated as:

$$x_{adv} = x + \varepsilon * sign(\nabla_x J(\Theta, x, y)) \tag{13}$$

where x_{adv} is the adversarial sample, x is the original input, y is the corresponding output after passing through the neural network, ε is the multiplier to ensure that the perturbation remains small, J is the loss function, and Θ are the parameters in the neural network.

Fig. 9 contrasts the signal strength of clean signals (Fig. 9(d)) with adversarial examples and additive Gaussian noise. Fig. 9(a) demonstrates how the input changes when ε is set to 0.005 and iterated for 10 loops. The system MAPE is 37.2 after reconstruction. When ε and iterations are increased, we observe that the MAPE increases to 43 as listed in Table IX. Different sub-groups contribute to the gradient in different directions during the iteration process. Thus, more effort is required before the adversarial example becomes impactful to all subgroups. Notably, the differences between original and adversarial signals are relatively small compared to the input signal's magnitude. This makes it challenging for the system to distinguish adversarial examples, potentially mistaking them for noise (Fig. 9(e)). However, the distribution of these differences is not random, unlike additive Gaussian noise. This unique distribution could be leveraged in future research to detect adversarial examples.

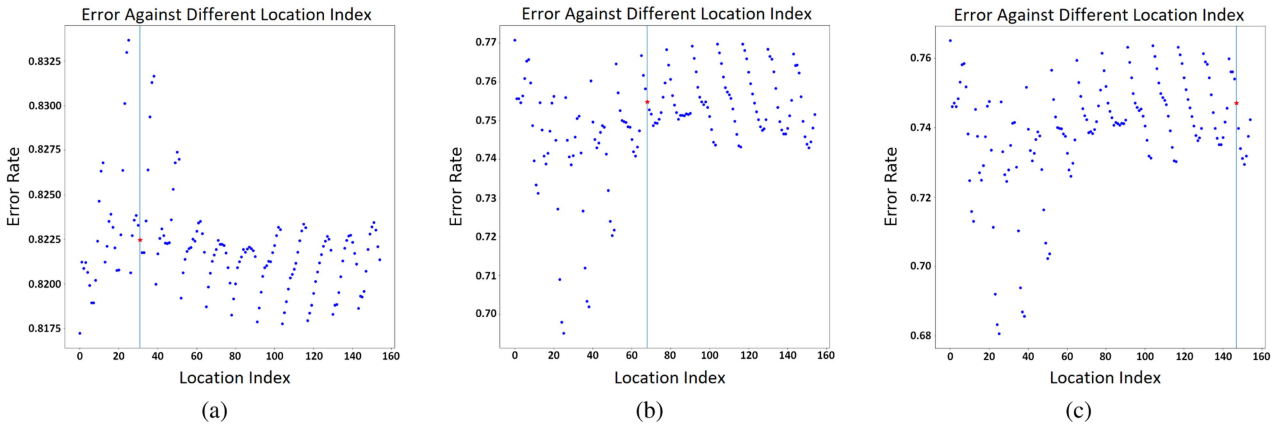

Fig. 8. Exhaustive search results for index having house information and a set of meter readings for multi-output model. Sub-captions indicate the data-id and the actual location index of the data. Vertical lines emphasize the specific index, while red stars signify the error rate associated with each location index during the attack.

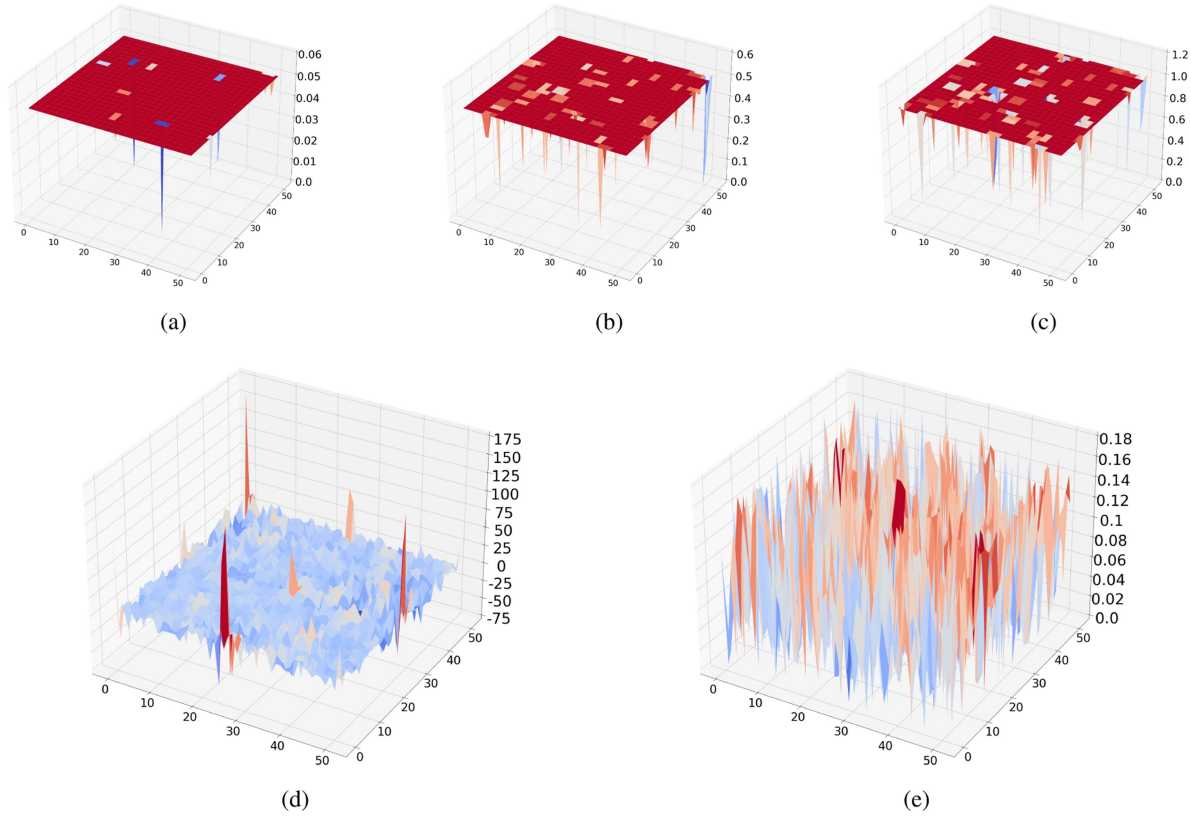

Fig. 9. Illustration of signal strength of adversarial samples, input signal and differences. x and y axis form the input matrix while z axis is the signal strength.

D. Cost and Trade-Off

This section delves into the additional costs and considerations associated with the proposed framework. By understanding these aspects, we can better gauge the trade-offs during implementation and identify the most suitable scenarios for our method.

The proposed approach is particularly advantageous in two situations:

1) *Adequate Computation Resources:* If there's ample computational power available, our method, which relies on parallel processing, becomes highly feasible.
2) *Fixed Data Rate from Various Subgroups:* Our method performs well when the system must handle a consistent

data rate from different subgroups, even if some of that data consists of zeros. This ensures efficient resource utilization and minimizes wastage.

To determine if a task is well-suited for our system, we can estimate its performance rank using:

$$Rank =\propto \frac{Task\ similarity}{(Feature\ sharing)(Task\ data\ rate)}. \quad (14)$$

Here:

- *Task Similarity:* This metric evaluates whether tasks fall within the same domain, for instance, predicting gas consumption across different subgroups.
- *Feature Sharing:* This quantifies the extent to which low-level features are shared among subgroups. A higher degree of feature sharing reduces the independence between subgroups.
- *Task Data Rate:* This pertains to the size of the subgroup. A larger subgroup size restricts the total number of subgroups that can be processed simultaneously, especially if computational resources are limited.

Given these factors, our method is especially beneficial when computational resources are limited but the input sources are numerous with a low data rate (e.g., smart meters and sensors). Alternatively, it is also effective when the input has a moderate data rate but fewer sources (e.g., small images).

Lastly, there's an inherent trade-off between accuracy and training cost. Our current model is designed to process data from all dwellings continuously. However, if only partial subgroup data is available, various subgroup combinations must be considered, which can affect both accuracy and computational demands.

V. Conclusion

This work introduced a novel approach to predict gas consumption of individual households, learning from subgroups within training data. In contrast to traditional single-output models, the proposed multi-output model can parallelize the prediction task to make an arbitrary number of simultaneous predictions. This prediction model, trained on weather conditions, house details, and gas consumption data from 155 dwellings over two years, demonstrated simplicity and resilience against external influences in extended period prediction. Our results highlighted several key findings:

- The proposed system delivers commendable prediction accuracy, while also offering the unique capability of making multiple concurrent predictions.
- The proposed system can make prediction over an extended period by using actual consumption data as a training guide while excluding it from the input matrix, thereby addressing the challenge of unavailable future data.
- The proposed system exhibits robustness against noisy and adversarial examples, ensuring reliable performance even in less-than-ideal conditions.
- The proposed system safeguards the private information that is used for training with a 99% success rate.

In essence, our proposed architecture lays a solid groundwork for subgroup-based learning, especially when navigating intricate design goals and the need for parallel processing of analogous tasks. Building on the transformation concept and subgroup learning strategy presented in this paper, future research could delve into integrating more intricate networks and value-based neural architectures, broadening the applicability of our approach to a wider array of applications.

References

[1] Y. Lu, S. Fang, G. Chen, T. Niu, and R. Liao, "Cyber-physical integration for future green seaports: Challenges, State of the Art and future prospects," *IEEE Trans. Ind. Cyber-Phys. Syst.*, vol. 1, pp. 21–43, 2023.

[2] J. Li, H. Li, T. Huang, L. Zheng, L. Ji, and S. Yin, "Model-free reinforcement learning economic dispatch algorithms for price-based residential demand response management system," *IEEE Trans. Ind. Cyber-Phys. Syst.*, vol. 1, pp. 123–135, 2023.

[3] Q. Lü et al., "Privacy-preserving decentralized dual averaging for online optimization over directed networks," *IEEE Trans. Ind. Cyber-Phys. Syst.*, vol. 1, pp. 79–91, 2023.

[4] "Smart meter," Accessed: Sep. 3, 2024. [Online]. Available: https://en.wikipedia.org/wiki/Smart_meter

[5] J. G. Jetcheva, M. Majidpour, and W.-P. Chen, "Neural network model ensembles for building-level electricity load forecasts," *Energy Buildings*, vol. 84, pp. 214–223, 2014.

[6] W. Kong, Z. Y. Dong, Y. Jia, D. J. Hill, Y. Xu, and Y. Zhang, "Short-term residential load forecasting based on LSTM recurrent neural network," *IEEE Trans. Smart Grid*, vol. 10, no. 1, pp. 841–851, Jan. 2019.

[7] A. Kell, A. S. McGough, and M. Forshaw, "Segmenting residential smart meter data for short-term load forecasting," in *Proc. 9th Int. Conf. Future Energy Syst.*, 2018, pp. 91–96.

[8] X. Hu and B. Sikdar, "Sub-group based machine learning for gas consumption prediction," in *Proc. IEEE Asia-Pacific Conf. Comput. Sci. Data Eng.*, 2021, pp. 1–6.

[9] A. Almalaq and G. Edwards, "A review of deep learning methods applied on load forecasting," in *Proc. IEEE Int. Conf. Mach. Learn. Appl.*, 2017, pp. 511–516.

[10] A. Ahmad, N. Javaid, M. Guizani, N. Alrajeh, and Z. A. Khan, "An accurate and fast converging short-term load forecasting model for industrial applications in a smart grid," *IEEE Trans. Ind. Inform.*, vol. 13, no. 5, pp. 2587–2596, Oct. 2017.

[11] T.-Y. Kim and S.-B. Cho, "Predicting residential energy consumption using CNN-LSTM neural networks," *Energy*, vol. 182, pp. 72–81, 2019.

[12] T. Le, M. T. Vo, B. Vo, E. Hwang, S. Rho, and S. W. Baik, "Improving electric energy consumption prediction using CNN and Bi-LSTM," *Appl. Sci.*, vol. 9, no. 20, 2019, Art. no. 4237.

[13] M. Tauber, F. Skopik, T. Bleier, and D. Hutchison, "A self-organising approach for smart meter communication systems," in *Proc. 7th IFIP TC 6 Int. Workshop Self-Organizing Syst.*, 2013, pp. 169–175.

[14] Z. Li, T. J. Oechtering, and M. Skoglund, "Privacy-preserving energy flow control in smart grids," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, 2016, pp. 2194–2198.

[15] I. Yilmaz and A. Siraj, "Avoiding occupancy detection from smart meter using adversarial machine learning," *IEEE Access*, vol. 9, pp. 35411–35430, 2021.

[16] M. Gönen, S. Khan, and S. Kaski, "Kernelized Bayesian matrix factorization," in *Proc. Int. Conf. Mach. Learn.* 2013, pp. 864–872.

[17] C. Li, F. Wei, W. Dong, X. Wang, Q. Liu, and X. Zhang, "Dynamic structure embedded online multiple-output regression for streaming data," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 2, pp. 323–336, Feb. 2019.

[18] Y. Huang and T. Zhong, "Multitask learning for neural generative question answering," *Mach. Vis. Appl.*, vol. 29, no. 6, pp. 1009–1017, 2018.

[19] K. Palasundram, N. M. Sharef, K. A. Kasmiran, and A. Azman, "SEQ2SEQ++: A multitasking-based Seq2seq model to generate meaningful and relevant answers," *IEEE Access*, vol. 9, pp. 164949–164975, 2021.

[20] I. Misra, A. Shrivastava, A. Gupta, and M. Hebert, "Cross-stitch networks for multi-task learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 3994–4003.

[21] S. Vandenhende, S. Georgoulis, B. De Brabandere, and L. Van Gool, "Branched multi-task networks: Deciding what layers to share," in *Proc. Brit. Mach. Vis. Conf.*, 2020.

[22] S. Ruder, J. Bingel, I. Augenstein, and A. Søgaard, "Latent multi-task architecture learning," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, pp. 4822–4829.

[23] P. Guo, C.-Y. Lee, and D. Ulbricht, "Learning to branch for multi-task learning," in *Proc. Int. Conf. Mach. Learn.* 2020, pp. 3854–3863.

[24] L. Wei, W. Tian, E. A. Silva, R. Choudhary, Q. Meng, and S. Yang, "Comparative study on machine learning for urban building energy analysis," *Procedia Eng.*, vol. 121, pp. 285–292, 2015.

[25] Y. Karadede, G. Ozdemir, and E. Aydemir, "Breeder hybrid algorithm approach for natural gas demand forecasting model," *Energy*, vol. 141, pp. 1269–1284, 2017.

[26] M. Akpinar, M. F. Adak, and N. Yumusak, "Day-ahead natural gas demand forecasting using optimized abc-based neural network with sliding window technique: The case study of regional basis in Türkiye," *Energies*, vol. 10, no. 6, 2017, Art. no. 781.

[27] B. de Keijzer et al., "Forecasting residential gas consumption with machine learning algorithms on weather data," *E3S Web Conf.*, vol. 111, 2019, Art. no. 05019.

[28] L. Nie et al., "Network traffic prediction in industrial Internet of Things backbone networks: A multitask learning mechanism," *IEEE Trans. Ind. Informat.*, vol. 17, no. 10, pp. 7123–7132, Oct. 2021.

[29] J. P. Wang, Y. K. Shi, W. S. Zhang, I. Thomas, and S. H. Duan, "Multitask policy adversarial learning for human-level control with large state spaces," *IEEE Trans. Ind. Inform.*, vol. 15, no. 4, pp. 2395–2404, Apr. 2019.

[30] C. Hong, J. Yu, J. Zhang, X. Jin, and K.-H. Lee, "Multimodal face-pose estimation with multitask manifold deep learning," *IEEE Trans. Ind. Inform.*, vol. 15, no. 7, pp. 3952–3961, Jul. 2019.

[31] "Pecan Street Inc.," Accessed: Dec. 30, 2016. [Online]. Available: https://dataport.pecanstreet.org/

[32] "Austin weather," Kaggle, 2017. Accessed: Jul. 2, 2020. [Online]. Available: https://www.kaggle.com/grubenm/austin-weather

[33] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[34] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *Proc. IEEE Symp. Secur. Privacy*, 2017, pp. 3–18.

[35] H. Qiu, Q. Zheng, T. Zhang, M. Qiu, G. Memmi, and J. Lu, "Toward secure and efficient deep learning inference in dependable IoT systems," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3180–3188, Mar. 2021.