Personalized Home Energy Management Using Non-Intrusive Load Monitoring and Human-in-the-Loop Reinforcement Learning

Ryan Persico[1], Yichen Yang[1], Rajit Gadh[2], Xiaofan Cui[1]
[1]Department of Electrical and Computer Engineering
[2]Department of Mechanical and Aerospace Engineering
University of California, Los Angeles
Los Angeles, CA, USA
rpersico@g.ucla.edu, cuixf@seas.ucla.edu

***Abstract*—The increasing adoption of electric vehicles, smart appliances, and residential energy storage systems calls for effective demand-side management (DSM) strategies; however, existing approaches are costly, intrusive, and difficult for users to configure, resulting in low adoption. This paper proposes a personalized DSM framework that integrates Non-Intrusive Load Monitoring (NILM), deep reinforcement learning, and a large language model (LLM) interface. The framework disaggregates whole-home smart-meter data into appliance-level consumption, employs a tunable Deep Q-Network (DQN) to schedule flexible loads under dynamic pricing while accounting for user habits, and leverages natural language interaction to capture evolving user preferences. This integration enables cost-aware, habit-preserving, and user-friendly household energy optimization. Evaluated on public datasets, the proposed NILM model achieves up to a 37.2 % reduction in mean absolute error compared with state-of-the-art methods, while the DQN scheduler reduces electricity cost by up to 24.6 % under Los Angeles time-of-use pricing without compromising user preferences.**

***Index Terms*—Demand side management, Non-intrusive load monitoring, Human-in-the-loop, Deep reinforcement learning, Large language models, Smart homes**

I. Introduction

Residential electrification is accelerating through increased adoption of electric vehicles, smart appliances, and home energy storage systems. This trend raises peak demand and motivates Home Energy Management Systems (HEMSs) that can shift or coordinate household loads to reduce electricity cost under time-varying prices [1]–[3].

A practical challenge is that effective demand-side management (DSM) decisions are inherently appliance- and session-specific (e.g., shifting the start time of a dishwasher cycle), whereas most residential settings only provide aggregate smart-meter measurements. Non-Intrusive Load Monitoring (NILM) addresses this observability gap by disaggregating whole-home power consumption into appliance-level usage using only the aggregate meter [4].

Existing DSM approaches largely rely on classical optimization frameworks, such as mixed-integer linear programming and nonlinear programming [5]. For example, residential demand response has been formulated as an optimal power flow (OPF) problem that coordinates household appliances while respecting network constraints [6]. However, most of these methods focus on electricity cost minimization while giving limited consideration to user comfort. Moreover, they often lack flexibility and adaptability, requiring users to manually specify detailed preferences and operational constraints. In practice, this process is tedious and impractical for everyday use, placing an excessive burden on the human in the loop.

Recent deep reinforcement learning (DRL) methods [7] have demonstrated the capability to jointly optimize economic objectives and comfort-related performance [8]. In parallel, advances in large language models (LLMs) enable intelligent parsing and interpretation of latent user behavioral preferences from natural interactions [9], offering new opportunities for more user-centric DSM solutions.

In this work, we aim to develop a low-cost and user-friendly home energy management approach that can adapt to user behavior while reducing electricity costs. First, a Subtask Gated Network (SGN) based NILM algorithm is used to extract appliance-level information directly from aggregated smart meter data, so no additional hardware or sub-metering is needed [10]. Based on this information, a Deep-Q-Network (DQN) based RL algorithm generates scheduling suggestions, presenting them to the user as recommendations. The user's acceptance or rejection is then used as feedback, allowing the system to gradually learn user preferences and balance the tradeoff between energy savings and comfort without requiring explicit preference inputs. Moreover, we integrate an LLM interface that translates natural-language user feedback into structured scheduling constraints, allowing the controller to adapt to evolving user priorities without manual reward redesign. The main contributions are:

- **End-to-end architecture:** an integrated NILM–RL–LLM framework that maps aggregate smart-meter data to user-adaptive, session-level DSM schedules.
- **Human-in-the-loop DSM:** A DQN scheduler balances energy cost and habit deviation while enforcing user preferences via calendar constraints. An LLM translates natural-language requests into structured device-

979-8-3315-8138-1

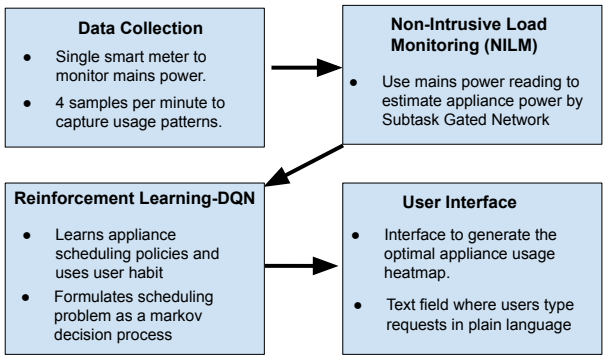


Fig. 1. Overall architecture of the proposed HEMS with four interconnected layers.

time updates that modify the feasible action set during scheduling.

The remainder of the paper describes the NILM-based session extraction, the DQN algorithm, the LLM module, and experimental results. The source code, as well as the Graphic User Interface (GUI) are publicly available [11].

II. Dataset and System Design

The proposed HEMS leverages the REFIT dataset [12] as the primary data source for NILM model development. REFIT contains appliance-level and aggregate mains measurements from 20 UK households, with an 8-second sampling rate over more than two years. In this work, we focus NILM training and evaluation on REFIT House 2, a two-story single-family house with three bedrooms. We use a time-based split (disjoint time ranges) for training, validation, and testing to avoid leakage across correlated daily routines. For DSM, we focus on shiftable, cycle-based appliances (e.g., dishwasher and washer) that are naturally represented as discrete operating sessions for start-time shifting. High impact loads such as HVAC systems and water heaters are not included because they are not submetered with consistent ground truth in the selected REFIT home, preventing reliable training and evaluation within our current dataset setup.

The NILM layer provides appliance-level usage traces inferred from aggregate mains data. These traces are aggregated into device sessions (contiguous ON intervals), which form the input to the DSM scheduling layer. The scheduler generates an optimized plan for the following day by shifting flexible sessions to minimize cost while preserving user habits through the LLM input and respecting preference calendars. In this way, the system closes the loop: NILM extracts fine-grained load information and reinforcement learning converts it into actionable schedules. The LLM interface personalizes this loop by translating natural language feedback into structured constraints, ensuring that optimized schedules align with evolving user preferences and comfort requirements. The overall architecture of interconnected layers is shown in Fig. 1.

III. NILM Module: SGN-based Disaggregation for Session Extraction

In our framework, NILM is a supporting module whose primary purpose is to enable robust extraction of discrete appliance sessions for downstream DSM scheduling. We build on the SGN concept from Shin et al. [10] due to its ability to suppress spurious predictions during OFF periods, which improves detection for session-based devices. SGN consists of a shared temporal encoder with regression and ON-probability heads. Their outputs are combined via a gate to suppress false activations, improving stability for downstream scheduling. The model architecture is shown in Fig. 2.

A. Implementation and Training Procedure

We train using REFIT House 2, with mains and appliance channels resampled to 15 s and window length $L{=}512$. Because the downstream controller operates on discrete sessions, our training procedure is designed to address severe ON/OFF imbalance and reduce fragmentation from spurious short activations near boundaries. We employ a three-stage training with biased mini-batch sampling:

- **Stage 1: ON-focused regressor Pretraining** We sample ON-rich windows and optimize a masked Huber loss over active periods to learn realistic power magnitudes without the regressor being dominated by OFF samples.
- **Stage 2: OFF/transition-focused classifier Pretraining** We sample OFF-rich windows and oversample ON/OFF transitions to improve boundary detection and reduce false positives that would otherwise create many short, spurious sessions.
- **Stage 3: Alternating Joint Training** We alternate ON-biased and OFF-biased mini-batches to jointly optimize both heads while preserving OFF-period suppression and stable transitions.

Joint optimization uses a weighted sum of regression and classification objectives:

$$\mathcal{L} = \alpha_{\text{on}}\mathcal{L}_{\text{reg@on}} + \alpha_{\text{off}}\mathcal{L}_{\text{reg@off}} + \alpha_{\text{reg}}\mathcal{L}_{\text{raw}} + \beta_{\text{cls}}\mathcal{L}_{\text{cls}},$$

Here, $\mathcal{L}_{\text{reg@on/off}}$ are masked regression terms for active/inactive periods, $\mathcal{L}_{\text{raw}}$ regularizes the ungated output, and $\mathcal{L}_{\text{cls}}$ is a focal loss mitigating ON/OFF imbalance. The coefficients $\alpha_{\text{on}}, \alpha_{\text{off}}, \alpha_{\text{reg}}$, and β_{cls} denote the respective weights of each term, ensuring balanced optimization and co-adaptation between regression and classification tasks.

B. Inference and Session Extraction

At inference, the SGN processes sliding windows of $L{=}512$ samples (15 s resolution), producing one prediction per time step. For deployment, we use the *hard-gated* output as the appliance estimate. The gating decision is obtained by applying a hysteresis rule to the classification probabilities, with separate ON/OFF thresholds and a minimum hold time to prevent rapid switching. This produces temporally consistent appliance-level predictions suitable for downstream DSM control.

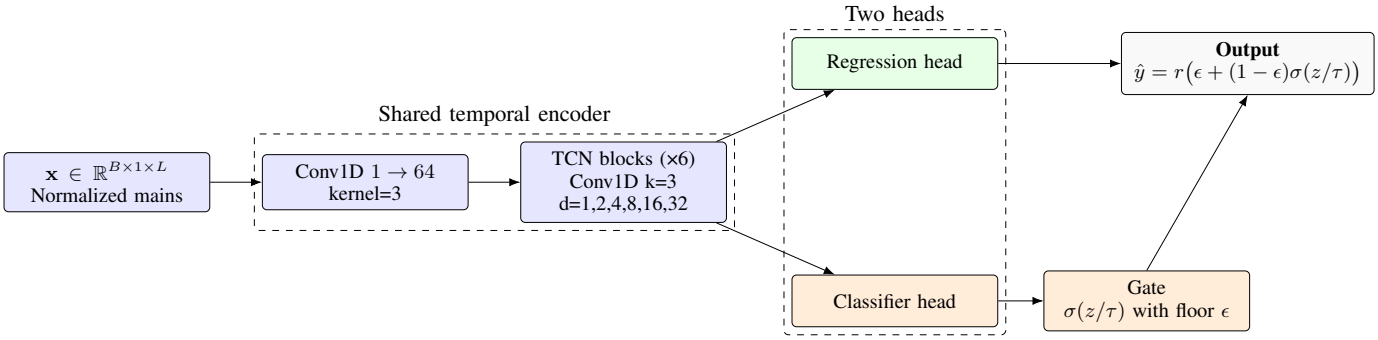


Fig. 2. Subtask-Gated Network (SGN) for NILM. The shared temporal encoder (blue) feeds regression (green) and classification (orange) heads; their outputs are merged through a gate into the final prediction $\hat{y}$.

C. Practical differences from standard SGN

Relative to a straightforward end-to-end SGN training setup, our pipeline emphasizes (i) a stage-wise ON/OFF curriculum to mitigate imbalance, (ii) transition oversampling to stabilize ON/OFF boundaries, and (iii) hysteresis-based hard gating with a minimum hold time to reduce fragmented activations. These choices are motivated by the downstream requirement of reliable session boundary extraction for scheduling rather than maximizing disaggregation accuracy alone, as typical NILM models do.

IV. DSM Optimization with Session-Shift DQN

We frame DSM as a session-shifting problem: each appliance usage session (a contiguous ON interval) is treated as a non-interruptible cycle whose primary control decision is *start time*. We therefore model controllable devices with a binary occupancy representation over time slots, while computing cost using the session power profile under time-varying prices. Each episode iterates through *all sessions sequentially*, forming a multi-step Markov Decision Process (MDP), updating device-level occupancy as sessions are placed.

Our scheduling agent is implemented as a DQN, inspired by the work of Mnih et al. [13] and Wang et al. [14], which demonstrated the effectiveness of deep reinforcement learning for sequential decision-making. Here, the DQN estimates action-values over feasible offsets, enabling the agent to balance energy cost, habit deviation, and user preferences within a unified learning framework. We use a fully connected Q-network with two hidden layers (256 and 128 units) with ReLU activations and Dropout(0.2).

A. Environment and Feasible Actions

Let N be the number of devices and T the horizon (time slots per day). For device d, session s is defined by a start index s_0 and a length L extracted from the historical profile. We define the offset set $\mathcal{K} = \{-K, \ldots, +K\}$. Feasibility for a given $k \in \mathcal{K}$ enforces: (i) no overlap with other sessions of the same device, (ii) respect hard-calendar slots.

B. State, Action, and Reward

For the currently selected (device, session) pair, the state concatenates scalar session descriptors with per-offset normalized components and a feasibility mask with $\dim(s) = 2+4A$. Here $A = 2K+1$ corresponds to the fixed ordered offset list $[-K, \ldots, 0, \ldots, +K]$; the mask marks offsets feasible at the current step after hard-constraint and overlap filtering. Each action a is an *index* $a \in \{1, \ldots, A\}$ selecting an *offset* $k \in \mathcal{K}$. The environment yields a step with reward

$$r \;=\; -\big(w_{\text{cost}}\, c_a \;+\; w_{\text{habit}}\, h_a \;+\; w_{\text{pref}}\, p_a\big), \qquad (1)$$

where:

- c_a is the *normalized energy cost* of placing the session at $s_0 + k$ given price and device power schedule (min–max normalization across feasible k),
- h_a is the *normalized habit deviation*, defined as $|k|$ divided by the maximum shift K,
- p_a is the *fraction of slots* in the placed session that violate the soft preference calendar.

Invalid actions receive a fixed penalty, and after valid actions we update the occupancy mask.

C. LLM-Enabled Preference Translation

To enable user-adaptive scheduling without manual reward redesign, we include a natural language interface that converts free-form user feedback into structured constraints consumed by the session-shift optimizer. The LLM operates as a translation layer: it maps a user text input (e.g., "Please do not run the Dishwasher or Kettle from 4 Pm to 6 Pm.") into a machine-readable command that updates device-time calendar constraints, which is then applied during schedule generation.

1) Natural language to structured command: Given a user input string, the LLM is prompted to output a single JSON object with the schema:

```
{
  "devices": [d1, ..., dk],
  "action":  "turn_on" | "turn_off",
  "time":    "hs, he"
}
```

Here `devices` must be selected from a fixed device vocabulary used by the HEMS front-end (e.g., `["Fridge-Freezer","Washing Machine","Dishwasher","Microwave"]`), and time is expressed in 24-hour time with integer hours $h_s, h_e \in [0, 23]$. In our implementation, we call `gpt-4` with temperature `0.2`.

2) Mapping commands to calendar constraints: The structured command is converted into a per-device, per-slot calendar over a daily horizon. The calendar entries encode constraints using a ternary representation:

$$\texttt{hard\_calendar}[d, t] \in \{-1, 0, 1\},$$

where -1 denotes unconstrained (flexible), 0 denotes *must-off*, and 1 denotes *must-on*. For an action `turn_off`, the specified interval is marked as must-off; for `turn_on`, it is marked as must-on. If multiple commands overlap, the most recent update takes precedence.

3) Effect on the session-shift optimization: The session-shift DQN consumes calendar information through the hard calendar input, which is converted into a feasibility mask m_a over offsets. In our implementation, the LLM affects scheduling by *modifying the feasible action set*: offsets that would violate must-off/must-on intervals are masked out and cannot be selected. This realizes user intent as quantitative constraints within the optimization, without changing the reward function itself.

4) Illustrative example: As an example, a user may enter: *"Please do not run the washing machine after 8 pm."* The LLM translates this request into the JSON command

```
{
  "devices":["Washing Machine"],
  "action":"turn_off",
  "time":"20, 0"
}
```

which blocks the interval 20:00–0:00 for the specified device. Without the constraint, the optimizer shifts sessions primarily to avoid peak price periods. With the constraint enabled, the optimizer produces a schedule that remains cost-aware while respecting the user-specified blocked interval. This additional mode of user input allows for personalized daily scheduling utilizing the existing framework.

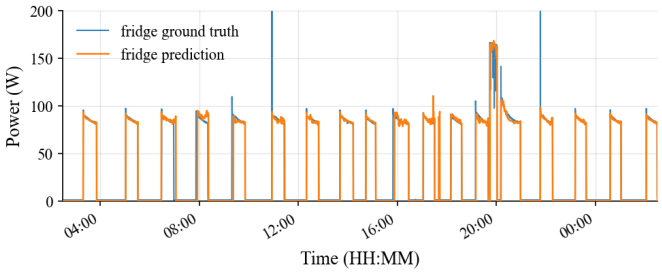
D. Training and Evaluation

Each episode resets device-level occupancy and iterates through all (device, session) pairs in chronological order, building the state and mask, selecting an offset with masked ϵ-greedy, stepping, and storing (s, a, r, s', m, m'). Transitions are sampled from a replay buffer and optimized with Adam using an MSE loss. After training, we perform a daily rollout by sequentially placing all sessions via masked $\arg\max$ to form a schedule, and report (i) total energy cost, (ii) mean habit deviation $|k|/K$, and (iii) preference-violation rate, each compared to a no-shift baseline.

- **Cost:** $\sum_t \sum_d a_{d,t}^{\text{DQN}} \, \text{Power}_{d,t} \, \text{Price}_t$ vs. baseline.
- **Habit deviation:** $\sum_d \sum_{\text{sessions}} |k|/K$ (baseline $= 0$).

TABLE I
PERFORMANCE OF SGN ON REFIT HOUSE 2.

Appliance	MAE (W)	F1	NDE
Dishwasher	4.06	0.962	0.0077
Fridge	4.99	0.947	0.0039
Kettle	5.34	0.828	0.0295
Washer	7.08	0.931	0.0165
Overall	5.37	0.917	0.0066



(a) Fridge

Fig. 3. NILM disaggregation results on REFIT House 2 showing SGN predictions versus ground truth.

- **Preference penalty:** sum of preference violation over all placed sessions.

V. RESULTS

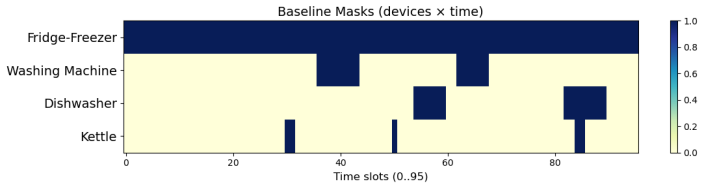
A. NILM Performance

Table I reports the disaggregation accuracy of the SGN on a held-out time segment from REFIT House 2 data across four appliances. We report three standard NILM metrics the MAE, the F1 score which evaluates ON/OFF state detection by combining precision and recall, which is particularly important for short-duration or binary appliances, and normalized disaggregation error (NDE). Together, these metrics capture both magnitude fidelity and event-level accuracy. The model achieves an overall MAE of 5.37 W, an F1 score of 0.917, and an NDE of 0.0066 shown in Table I.

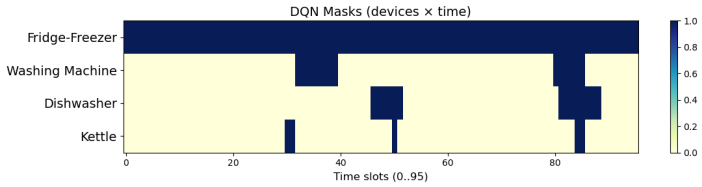
Figure 3 shows sample disaggregation traces for the fridge. The SGN predictions closely follow ground truth consumption, with the hard-gated output suppressing spurious activations during idle periods.

B. DQN-Based Demand Side Management

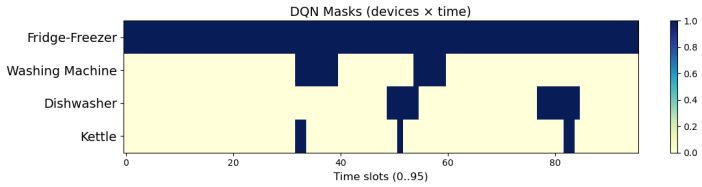
Example schedules. Figure 4 illustrates a representative daily schedule and its corresponding optimized schedule with and without LLM input. Enabling the LLM-derived hard-calendar constraint prevents the washing machine from being scheduled during the blocked interval specified by the user request (20:00–0:00 in our example) with zero blocked-interval violations under the LLM constraint. The optimized versions shift flexible loads out of costly peak hours while avoiding overlaps because sessions are typically nudged by only a few 15-minute slots, so recommendations remain realistic and usually acceptable. Table II aswell as Figure 4 summarizes a representative-day ablation comparing the baseline schedule



(a) Baseline



(b) Optimized (DQN)



(c) Optimized (DQN) With LLM input

Fig. 4. Example appliance schedules: (a) baseline, (b) DQN-optimized, (c) DQN-optimized schedule with user input example above. Where 1 represents an appliance's on-state and 0 its off-state. Time of Use energy cost based off of Los Angeles electric rates.

TABLE II
REPRESENTATIVE-DAY ABLATION COMPARING BASELINE (NO SHIFTING) AND DQN SCHEDULING WITHOUT CONSTRAINTS. METRICS REPORT SCHEDULING OUTCOMES AND THE RESULTING FEASIBLE ACTION SPACE.

	Base	**DQN**
Cost	4919.1	3712.1
Habit dev.	0.00	3.25
Feas. actions (of 17)	–	13.00
Mean $\|k\|$ (min)	0.0	5
Shift rate	0/8	7/8

(no shifting) and the unconstrained DQN schedule, and the DQN with the LLM-derived hard-calendar constraint. Relative to the baseline, the DQN substantially reduces cost down to 3712, and enabling the LLM constraint increases cost slightly up to 4084, reflecting a cost–constraint trade-off. The constraint also reduces the average number of feasible actions per decision step (from 13 feasible actions to approximately 10 actions out of 17).

User Behavior Simulation and Adaptive Evaluation. We validate the human-in-the-loop setting using a *Discrete-ChoiceUser* simulator [11] that models acceptance of suggested shifts via a softmax utility over habit discomfort, calendar violations, and inertia. The scheduler proposes daily shifts, the simulated user accepts or rejects the session shifts within. This forms a human-in-the-loop closed feedback loop. Over multiple weeks, the interaction converges to a stable repeated schedule while maintaining cost reductions, indicating that the controller can adapt under realistic acceptance behavior.

VI. CONCLUSION

This paper presented a user-friendly human-in-the-loop home energy management system that integrates NILM, a session-shift DQN scheduler, and an LLM interface that translates natural-language preferences into structured calendar constraints for DSM. Tested on REFIT, an open-access benchmark dataset, the proposed algorithm achieved up to 24.6% normalized cost reduction while preserving user habits via adaptive reward weighting. Future work will validate the approach across diverse households and extend it to thermostat-controlled loads and EV mobile energy storage to better capture whole-home demand flexibility.

REFERENCES

[1] B. Zhou, W. Li, K. W. Chan, Y. Cao, Y. Kuang, X. Liu, and X. Wang, "Smart home energy management systems: Concept, configurations, and scheduling strategies," *Renewable and Sustainable Energy Reviews*, vol. 61, pp. 30–40, 2016.

[2] P. Palensky and D. Dietrich, "Demand side management: Demand response, intelligent energy systems, and smart loads," *IEEE transactions on industrial informatics*, vol. 7, no. 3, pp. 381–388, 2011.

[3] J. S. Vardakas, N. Zorba, and C. V. Verikoukis, "A survey on demand response programs in smart grids: Pricing methods and optimization algorithms," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 152–178, 2014.

[4] A. Zoha, A. Gluhak, M. A. Imran, and S. Rajasegarar, "Non-intrusive load monitoring approaches for disaggregated energy sensing: A survey," *Sensors*, vol. 12, no. 12, pp. 16 838–16 866, 2012.

[5] A. R. Jordehi, "Optimisation of demand response in electric power systems, a review," *Renewable and sustainable energy reviews*, vol. 103, pp. 308–319, 2019.

[6] W. Shi, N. Li, X. Xie, C.-C. Chu, and R. Gadh, "Optimal residential demand response in distribution networks," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 7, pp. 1441–1450, 2014.

[7] J. R. Vázquez-Canteli and Z. Nagy, "Reinforcement learning for demand response: A review of algorithms and modeling techniques," *Applied energy*, vol. 235, pp. 1072–1089, 2019.

[8] A. A. Amer, K. Shaban, and A. M. Massoud, "Drl-hems: Deep reinforcement learning agent for demand response in home energy management systems considering customers and operators perspectives," *IEEE Transactions on Smart Grid*, vol. 14, no. 1, pp. 239–250, 2023.

[9] J. Liu, X. Yan, D. Li, G. Zhang, H. Gu, P. Zhang, T. Lu, L. Shang, and N. Gu, "Improving llm-powered recommendations with personalized information," in *Proceedings of the 48th International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '25. New York, NY, USA: Association for Computing Machinery, 2025, p. 2560–2565. [Online]. Available: https://doi.org/10.1145/3726302.3730211

[10] C. Shin, J. Joo, H. Song, H.-S. Lee, and I. Lee, "Subtask gated networks for non-intrusive load monitoring," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 1150–1157. [Online]. Available: https://doi.org/10.1609/aaai.v33i01.33011150

[11] R. Persico, Y. Yang, and X. Cui, "Nilm–dr hybrid hems pipeline (code and gui figures)," https://github.com/Rcpersico/DSM_NILM_PIPELINE, 2025.

[12] D. Murray, L. Stankovic, and V. Stankovic, "REFIT: Electrical Load Measurements (Cleaned)," 2017. [Online]. Available: https://pureportal.strath.ac.uk/en/datasets/refit-electrical-load-measurements-cleaned

[13] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[14] B. Wang, Y. Li, W. Ming, and S. Wang, "Deep reinforcement learning method for demand response management of interruptible load," *IEEE Transactions on Smart Grid*, vol. 11, no. 4, pp. 3146–3155, 2020.