Behavior Testing of Load Forecasting Models using BuildChecks

Yang Deng, Jiaqi Fan, Hao Jiang, Fang He, Dan Wang, Ao Li, Fu Xiao
The Hong Kong Polytechnic University
{yang2.deng,18081491d,18082566d,fangf.he,18070016r}@connect.polyu.hk
{dan.wang,linda.xiao}@polyu.edu.hk

ABSTRACT

In recent years, machine learning (ML) models have been widely developed for building systems. For example, a number of ML models have been developed to predict the load demand of a building. Current ML models commonly report snap-shot accuracy only. Practitioners have difficulties in understanding how a model behaves in *usage*, i.e., model accuracy may change during model usage. This raises concerns in the ML-model deployment.

In this paper, we propose BuildChecks, a *behavior testing methodology* to systematically evaluate building load forecasting ML models in usage. The challenge of developing such a methodology is to specify "what to evaluate", i.e., given a certain building load forecasting model, what tests we shall apply to this model. We categorize three *model-types* of the building load forecasting models and we propose three *in-usage concerns*. Our methodology specifies the *tests*, i.e., for each model-type, the in-usage concerns that should be tested. We develop an open-source BuildChecks *platform* to materialize our behavior testing methodology. The BuildChecks platform integrates the testing algorithms and four default real-world building datasets. We use BuildChecks to test the behaviors of two existing load forecasting models. As an example, while a ML model has high accuracy throughout all buildings, BuildChecks reports that in one building this ML-model has a cold start time of 45 days, yet in another building, the cold start time is three-fold greater, 141 days – this can lead to a delay in model usage.

CCS CONCEPTS

• **General and reference** → **Evaluation**.

KEYWORDS

Smart building, Machine learning, AI Evaluation

ACM Reference Format:
Yang Deng, Jiaqi Fan, Hao Jiang, Fang He, Dan Wang, Ao Li, Fu Xiao. 2022. Behavior Testing of Load Forecasting Models using BuildChecks. In *The Thirteenth ACM International Conference on Future Energy Systems (e-Energy '22), June 28–July 1, 2022, Virtual Event, USA*, 5 pages. https://doi.org/10.1145/3538637.3538841

e-Energy '22, June 28–July 1, 2022, Virtual Event, USA

ACM ISBN 978-1-4503-9397-3/22/06…$15.00
https://doi.org/10.1145/3538637.3538841

1 INTRODUCTION

Buildings are major energy consumers and carbon emitters in modern society [26]. In the US, buildings account for over 40% of total energy usage [2]. To better operate building systems and conserve energy, building load forecasting plays an important role. Recently, with the development of IoT and AI technologies, the building automation systems (BAS) have been transformed into information supported decision making systems. This provides ample opportunities to develop data-driven machine learning (ML) load forecasting models for building operation and control [10, 15, 17, 18].

Current studies on ML models emphasize on new model development under various contexts and application of ML algorithms [11, 18, 27]. We notice that current ML models commonly report snap-shot accuracy only. Practitioners have difficulties in understanding how a model behave *in-usage*, i.e., model accuracy may change during model usage. This raises concerns on deploying ML models in practice. There are studies on improving model accuracy during usage for an individual ML model; yet practitioners are more eager to understand and compare the behaviors of a wealth of models. In this work, we argue that we need to test the behaviors of building load forecasting models in-usage. Behavior testing (also known as black-box testing) is a software engineering concept that tests system capabilities by validating the input-output behavior, without the knowledge of the internal structure [5]. In simple, behavior testing defines organized tests (i.e., what should (or should not) be tested) for diverse scenarios. We, for the very first time, bring behavior testing to ML models in smart buildings.

The challenge of developing behavior testing is to define "what to evaluate", i.e., given a building load forecasting model, what tests we shall apply to this model. We categorize three *model-types* of the building load forecasting models: a same model-type indicates that the ML models have similar objectives and are expected to perform in comparison. We propose three *in-usage concerns*. We develop a *behavior testing methodology*, BuildChecks, which specifies the *tests*, i.e., for each model-type, the in-usage concerns that are meaningful to this model-type and should be tested. We develop a BuildChecks *platform* to realize our behavior testing methodology. This platform integrates four default building data sets and the algorithms for behavior testing. We evaluate our proposed BuildChecks by testing the behaviors of two existing building load forecasting models, HK-ICC [18] and London-Residental [11].

BuildChecks reports test results and how to use the results depends on users and is not in the scope of BuildChecks. Nevertheless, we illustrate two simple examples, one from the perspective of an ML model and one from the perspective of a data set: (1) BuildChecks tests the HK-ICC model with a set of retrain strategies and BuildChecks reports (Figure 3) that the the ADWIN detection algorithm [6] maintains high accuracy in usage for the HK-ICC model;

and (2) the EM dataset has a sub-period of four months where the region has a work-from-home (WFH) policy due to Covid-19. To serve this special period, a model with short cold-start time is important. BuildChecks can report the cold start time of ML models and our evaluation shows that the cold start time of a London-Residential model can be 42% to that of the HK-ICC model though the HK-ICC model has higher accuracy.

In summary, the contribution of our paper is three-fold: (1) we propose a behavior testing methodology to evaluate the ML models for building load forecasting (§2). To the best of our knowledge, we are the first to bring behavior testing, a software engineering concept into ML model testing in buildings; (2) we develop a BuildChecks platform to materialize our behavior testing methodology (§3); and (3) we use BuildChecks to test the behaviors of a number of models and we present two cases (§4) in the interest of space.

2 THE BUILDCHECKS METHODOLOGY

We define our behavior testing methodology from three organized aspects (Figure 1 shows the BuildChecks methodology):

Selection of in-usage concerns: As said, in-usage concerns refer to model accuracy change during usage. In-usage concerns have been heavily studied in the ML community [22, 32–34]. We select three typical ones and our methodology can be easily extended.

(1) Cold start [23, 29, 34]: History data needs to be accumulated so that an ML model can be trained to an acceptable accuracy. This time period is the cold start time. A shorter cold start shows that the ML model can meet the engineering purposes and be deployed at an (sometimes substantially) earlier time.
(2) Retrain strategy [7, 28, 32]: ML models face concept drifts [14] in usage and need to be retrained to maintain accuracy. There are different types of retrain strategies and it is important to figure out the suitable ones for a certain model.
(3) Catastrophic forgetting [19, 22, 33]: Many ML models have internal mechanism for updating the model parameters to learn new knowledge. They may forget existing learned knowledge when learning new ones.

Building load forecasting model categorization: ML models may not face the same in-usage concerns. For example, some models (e.g., the HK-ICC model [18]) are not designed with an internal model update mechanism and thus do not have the catastrophic forgetting concern. To determine what in-usage concerns to test for an ML model, we categorize ML models into three *model-types* which are widely-accepted in building automation studies [31].

(1) Short-term forecasting with snapshot designs (SSF): Short-term models are mostly studied and account for more than 84% of the building load forecasting models [9, 15].
(2) Short-term forecasting with online learning (SOF): There are short-term models with internal mechanisms to online update the model parameters [11, 20, 24]. Note that such online update mechanisms differ from the retrain strategy since online learning focus on local adjustment, yet the retrain strategy will apply all (or a majority of) history data.
(3) Midterm/long-term forecasting (MLF) [1, 4].

Defining tests: We define tests based on the in-usage concerns and the model-types, i.e., for a certain model-type, whether a certain in-usage concern should be or should not be tested.

Concerns in Usage / Model Types	Cold Start	Retrain Strategy	Catastrophic Forgetting
Short-term forecasting with snapshot design (SSF)	√	√	N/A
Short-term forecasting with online learning (SOF)	√	√	√
Middle\long-term forecasting (MLF)	×	√	√

√: It should be evaluated. ×: It is not meaningful to evaluated. N/A: Cannot evaluate.

Figure 1: The BuildChecks Methodology

(1) Tests for SSF models: clearly, cold start and retrain strategy should tested. The SSF model designs focus on generality, not on local adjustment. As such, the catastrophic forgetting concern cannot be tested for SSF models.
(2) Tests for SOF models: all in-usage concerns are to be tested.
(3) Tests for MLF models: it is not meaningful to test cold start for MLF models since they have long service time and the cold start period only has minimal influence.

3 THE BUILDCHECKS PLATFORM

We develop a platform to realize our behavior testing methodology.

Goals and BuildChecks design choices: The BuildChecks platform serves two types of users: (1) those with their own building data and (2) those without building data. Consequently, we have two design goals: (i) To integrate certain default building datasets into the platform to serve users without building data and (ii) To integrate testing algorithms on cold-start, retrain strategy, and catastrophic forgetting into the platform. In this paper, we omit other goals, e.g., performance, scalability, etc., since they are not directly related to realizing our methodology. The framework of our platform is shown in Figure 2.

In the building automation community, datasets are usually not open to public due to privacy concerns. BuildChecks should neither claim building data from users nor disclose its default building datasets during tests. To this end, our framework consists of a model testing layer and a building data layer to hold default building datasets. For users with their own data, the separation of the building data layer from the model testing layer makes it easy for the BuildChecks platform to be installed directly (without default datasets) in their own servers where they can simply insert their own data. For users without building data, they can submit their models to BuildChecks, and we report results without disclosing our default datasets.

We present the testing algorithms in the model testing layer.

- Cold start: We adopt the widely accepted stopping condition (defined by ASHRAE [30]) for the cold start period, i.e., the ML model accuracy should reach a threshold, CV-RMSE<30%. We find the cold start period through a binary-search.
- Retrain strategy: We integrate three retrain strategies into BuildChecks: (1) periodical retrain strategy; (2) retrain based on a error rate-based concept drift detection algorithm, Drift Detection Method (DDM) [12]; and (3) retrain based on a data distribution-based concept drift detection algorithm Adaptive Windowing (ADWIN) [6].
- Catastrophic forgetting: there are two standard online updating mechanisms for SOF and MLF, e.g., periodically online update for shallow ML models and online fine-tuning for

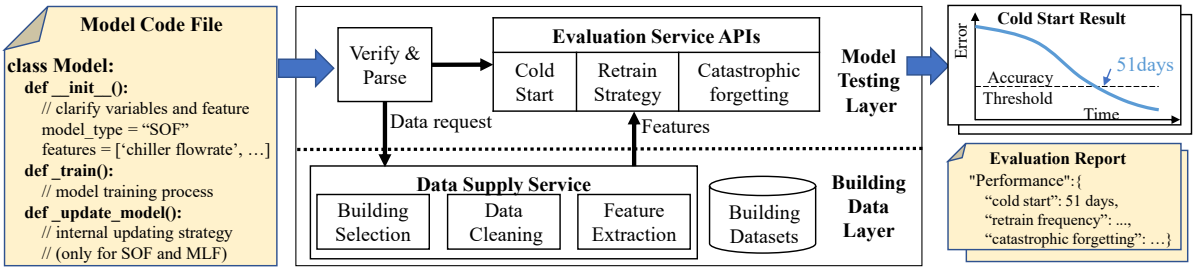

Figure 2: The framework of BuildChecks.

Dataset	Sites	Sampled rate	Chiller Number	Building Type	Length
EM	2	15min, 1h	3, 3	Commercial	1.5 years
HI	9	1h	4 to 8	Commercial	2 years
AD	5	1day	5	Residential	4 years
SZ	1	30mins	6	Data center	1 year

Table 1: Dataset specification

DNN models. Accordingly, we implement two statistic built-in functions for each of these categories.

BuildChecks implementation: We implement a prototype of the BuildChecks platform (Figure 2). We implement the building data layer using MySQL database. We integrate BuildChecks with four default datasets that cover diverse buildings. The basic characteristics of these datasets are shown in Table 1. We develop a data supply service module, which executes data request management, building dataset selection, as well as standard data preprocessing tasks. We comment that we also standardize the coding format for model sub-functions that are related to model evaluation.

4 CASE STUDIES

We use BuildChecks to evaluate the behaviors of two representative models of SOF and SSF type:

- The London-Residential model [11]: this is a SOF model. The model is based on LSTM neural network. More importantly, this model introduces adaptive buffering and tuning modules to handle contextual adaptation.
- The HK-ICC model [18]: this is an SSF model. It has an attention mechanism based on the Seq2Seq [25] neural network to extract the short-term dynamic temporal load pattern.

Our metrics are the **length** of cold start period [13], the **Mean Accuracy** under retrain strategy [3, 14], and the **Average forgetting** for catastrophic forgetting, which is defined as the difference between the accuracy when first learning a task, and the accuracy decay after training one or more additional tasks [8, 21].

4.1 The London-Residential model

4.1.1 Evaluation results on cold start. As described in London-Residential [11]), a base model would be trained at first, then the base model would be online updated during its runtime. Therefore, it is necessary to evaluate the cold start of the base model.

The cold start evaluation follows the prequential method [13], which is a general methodology to evaluate learning algorithms in streaming scenarios. As shown in Figure 3(a), the cold start of the model is 84 days, 58 days, 138 days and 51 days in EM_1H, EM_15min, HI (nine buildings) and SZ respectively. The model achieves shortest cold start period on SZ could due to that the load demand pattern in this data center dataset is relatively flat while compared to other three tested buildings. Moreover, we note that a shorter cold start does not mean greater accuracy in usage. The accuracy in usage is 27% lower in EM_1H while compared to in HI, although the cold start is shorter in EM_1H (84 days vs 138 days).

4.1.2 Evaluation results on retrain strategy. We discuss the different retrain strategies and compare the overall accuracy performance. The results are shown in Figure 3(b) and 3(c).

Periodically retrain. As shown in Figure 3(b), In EM_1H and SZ building, which shows the benefit of periodically retrain is moderate (within 5%). For EM_15min and HI, the model with periodic retrain can improve the accuracy by 21% and 36%, respectively.

Triggered retrain. The model can be retrained triggered by concept drift detection algorithms (DDM and ADWIN as introduced). In Figure 3(c), We observe in HI, the median accuracy of model retrained based on drift detection algorithms can improve the accuracy in use by 29% and 14% for ADWIN and DDM. The accuracy improvement in other three datasets is minor (within 6%). Particularly, there is a very slight accuracy decay in SZ data center.

We observe the benefits of retrain strategy in the model. The mean accuracy improvements are 7.3%, 11.6%, and 9.3% for the periodically retrain strategy, the ADWIN strategy, and the DDM strategy. The model has online update mechanisms. We note that the retrain strategies may overlap with the online updating mechanisms internal to a SOF model. Nevertheless, the main objective of BuildChecks is to test ML models and report results. It is beyond the scope of BuildChecks to explore such overlap.

4.1.3 Evaluation results on catastrophic forgetting. The evaluation process follows the online continual learning setting [21]. London-Residential was first trained to achieve CV-RMSE < 30% and then the model keeps updating following its internal updating mechanism.

Average forgetting. Figure 3(d) shows how much of the acquired knowledge the model has forgotten during the 12 times of model update (we set the update frequency is 10 days as defined in London-Residential [11]). During the first five updates, the forget ranges from -98 to 0 for all tested buildings. This reveals not only the model does not forget learned knowledge, but also enhances its prediction capacity. After that, London-Residential model performs differently among the datasets. The forget value is gradually increasing to over 378 in HI after model update 12 times, which illustrates the model is getting worse on the previous load context.

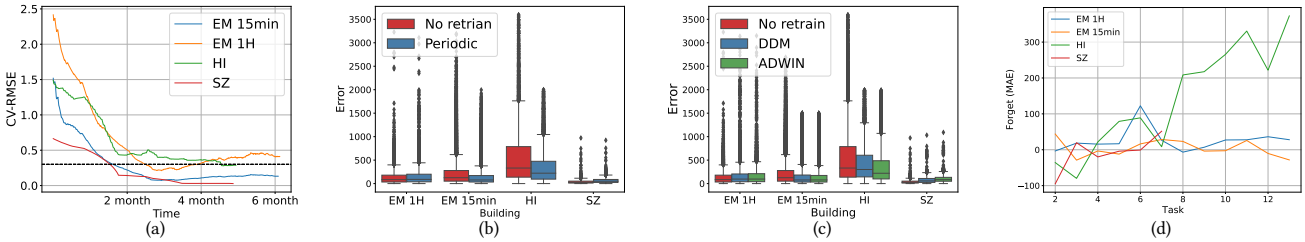

Figure 3: (a) The length of cold start period; (b) The Accuracy as retrained periodically; (c) The accuracy as retrained by two triggered strategies; (d) The forget measured by the end of each update.

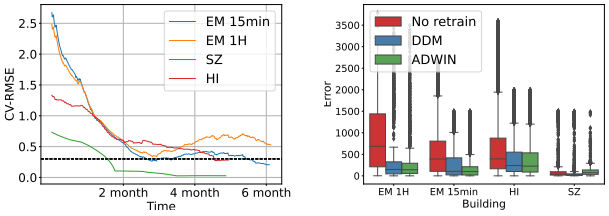

Figure 4: (a) The length of cold start period; (b) The accuracy as retrained by two triggered strategies.

4.1.4 Brief summary. Following BuildChecks, London-Residential was evaluated in all three behavior tests. For the cold start, the model has a relatively short cold start in the SZ data center building (within 5 weeks), while on the HI residential buildings is larger than 17 weeks. For the retrain strategy, BuildChecks illustrates that all the three tested retrain strategies can achieve moderate accuracy improvement (within 10% on average). For catastrophic forgetting, the model in HI is progressively worse over time. Yet there is no significant forgetting in other tested buildings.

4.2 The HK-ICC model

As a SSF model, cold start and retrain strategy are evaluated.

Evaluation results on cold start. We can observe in Figure 4(a) that the cold start is 93 days, 85days, 141 days and 45 days for the tested datasets (SZ is still the shortest). In the two buildings in EM, the curves have a rebound tendency in the third month, we infer the cold start of this model could be effected by the COVID-19 WFH (work from home) in this period.

Evaluation results on retrain strategy. All the retrain strategies perform well in the HK-ICC evaluation. We only discuss the trigger-based strategy since it can achieve greater accuracy than periodical strategy. The results are shown in Figure 4(b). The medium error of the ADWIN-based retrain is 167, 109, 244, and 86 for the four datasets respectively, and the performance is slightly better than DDM (about 2%). On the other hand, the proportion of outliers (error rate > 95%) in DDM based retrain is less than ADWIN (about 19%), which shows the robustness of DDM in our test buildings.

Brief summary. HK-ICC also has a shorter cold start time in the SZ data center buildings. And all the retrain strategies can bring in much greater accuracy improvement (22% on average). The report shows that the triggered strategy is the best choice for HK-ICC, especially in EM buildings.

5 RELATED WORK

Building load forecasting is important for efficient building operations [31]. Recently, many ML models have been developed [31]. As compared to the physical and statistic models, ML models are known to be less explainable. Moreover, ML models are commonly evaluated by snap-short accuracy only. Thus, practical adoption is still slow. This paper differs from the efforts in explaining [16] the behaviors of an ML model; instead, we propose to standardize the evaluation on ML in-usage concerns; thus allowing selection and comparison of ML models. The idea was sparked by the behavior testing concept from software engineering.

In-usage concerns of ML models have been studied in ML community, e.g., in recommendation systems, question answering system, etc. It is known that the accuracy may change due to a number of reasons. To warm up the cold start process, embedding techniques have been proposed [34]. Retrain strategies have been heavily studied, e.g., an advanced one is a meta-learning strategy [32]. Replay of historical samples has been proposed [22, 33] to avoid catastrophic forgetting. We borrow these in-usage concerns into the definition of our behavior testing methodology.

6 CONCLUSION

In this paper, we presented BuildChecks, a behavior testing methodology and an associated platform to evaluate the behaviors of ML-based building load forecasting models. BuildChecks can output reports on the model in-usage concerns such as cold start, retrain strategy, and catastrophic forgetting. BuildChecks complements the understanding of the ML models from the perspective of model accuracy to a richer context. We use BuildChecks to test the behaviors of two models in different buildings. For the London-Residential model, BuildChecks reports that a retrain strategy is meaningful for the HI building (e.g., with 36% improvement), yet the improvement is immaterial for the other three building datasets.

We comment that BuildChecks is designed to provide guidelines on the behavior testing process so that models can be comprehensively compared (it is less an objective to explain behaviors). Clearly, there are different angles to *define* the testing methodology. It is our future work to investigate other testing criteria.

7 ACKNOWLEDGEMENTS

Dan Wang's work is supported by the National Key Research and Development Program of China under Grant No. 2020YFE0200500 and by GRF 15210119, 15209220, 15200321, ITF-ITSP ITS/070/19FP, CRF C5026- 18G, C5018-20G, PolyU 1-ZVPZ.

REFERENCES

[1] Rahul Kumar Agrawal, Frankle Muchahary, and Madan Mohan Tripathi. 2018. Long term load forecasting with hourly predictions based on long-short-term-memory networks. In *2018 IEEE Texas Power and Energy Conference (TPEC)*. IEEE, 1–6.

[2] Kadir Amasyali and Nora M El-Gohary. 2018. A review of data-driven building energy consumption prediction studies. *Renewable and Sustainable Energy Reviews* 81 (2018), 1192–1205.

[3] Lucas Baier, Vincent Kellner, Niklas Kühl, and Gerhard Satzger. 2021. Switching Scheme: A Novel Approach for Handling Incremental Concept Drift in Real-World Data Sets. In *Proceedings of the 54th Hawaii International Conference on System Sciences*. 990.

[4] Roberto Baviera and Michele Azzone. 2021. Neural network middle-term probabilistic forecasting of daily power consumption. *Journal of Energy Markets* 14, 1 (2021).

[5] Boris Beizer. 1995. *Black-box testing: techniques for functional testing of software and systems*. John Wiley & Sons, Inc.

[6] Albert Bifet and Ricard Gavalda. 2007. Learning from time-changing data with adaptive windowing. In *Proceedings of the 2007 SIAM international conference on data mining*. SIAM, 443–448.

[7] Tao Chen. 2019. All versus one: An empirical comparison on retrained and incremental machine learning for modeling performance of adaptable software. In *2019 IEEE/ACM 14th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS)*. IEEE, 157–168.

[8] Matthias Delange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Ales Leonardis, Greg Slabaugh, and Tinne Tuytelaars. 2021. A continual learning survey: Defying forgetting in classification tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2021).

[9] Cheng Fan, Fu Xiao, and Shengwei Wang. 2014. Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques. *Applied Energy* 127 (2014), 1–10.

[10] Cheng Fan, Fu Xiao, and Yang Zhao. 2017. A short-term building cooling load prediction method using deep learning algorithms. *Applied energy* 195 (2017), 222–233.

[11] Mohammad Navid Fekri, Harsh Patel, Katarina Grolinger, and Vinay Sharma. 2021. Deep learning for load forecasting with smart meter data: Online adaptive recurrent neural network. *Applied Energy* 282 (2021), 116177.

[12] Joao Gama, Pedro Medas, Gladys Castillo, and Pedro Rodrigues. 2004. Learning with drift detection. In *Brazilian symposium on artificial intelligence*. Springer, 286–295.

[13] Joao Gama, Raquel Sebastiao, and Pedro Pereira Rodrigues. 2013. On evaluating stream learning algorithms. *Machine learning* 90, 3 (2013), 317–346.

[14] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)* 46, 4 (2014), 1–37.

[15] Narendhar Gugulothu and Easwar Subramanian. 2019. Load Forecasting in Energy Markets: An Approach Using Sparse Neural Networks. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems*. 403–405.

[16] Fred Hohman, Haekyu Park, Caleb Robinson, and Duen Horng Polo Chau. 2019. S ummit: Scaling deep learning interpretability by visualizing activation and attribution summarizations. *IEEE transactions on visualization and computer graphics* 26, 1 (2019), 1096–1106.

[17] Aria Jozi, Tiago Pinto, and Zita Vale. 2022. Contextual learning for energy forecasting in buildings. *International Journal of Electrical Power & Energy Systems* 136 (2022), 107707.

[18] Ao Li, Fu Xiao, Chong Zhang, and Cheng Fan. 2021. Attention-based interpretable neural network for building cooling load prediction. *Applied Energy* 299 (2021), 117238.

[19] Xilai Li, Yingbo Zhou, Tianfu Wu, Richard Socher, and Caiming Xiong. 2019. Learn to grow: A continual structure learning framework for overcoming catastrophic forgetting. In *International Conference on Machine Learning*. PMLR, 3925–3934.

[20] Fan Liang, William Grant Hatcher, Guobin Xu, James Nguyen, Weixian Liao, and Wei Yu. 2019. Towards online deep learning-based energy forecasting. In *2019 28th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 1–9.

[21] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. 2022. Online continual learning in image classification: An empirical survey. *Neurocomputing* 469 (2022), 28–51.

[22] Fei Mi, Xiaoyu Lin, and Boi Faltings. 2020. Ader: Adaptively distilled exemplar replay towards continual learning for session-based recommendation. In *Fourteenth ACM Conference on Recommender Systems*. 408–413.

[23] Konstantinos Pliakos, Seang-Hwane Joo, Jung Yeon Park, Frederik Cornillie, Celine Vens, and Wim Van den Noortgate. 2019. Integrating machine learning into item response theory for addressing the cold start problem in adaptive learning systems. *Computers & Education* 137 (2019), 91–103.

[24] M Pratapa Raju and A Jaya Laxmi. 2020. IOT based online load forecasting using machine learning algorithms. *Procedia Computer Science* 171 (2020), 551–560.

[25] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems* 27 (2014).

[26] Diana Ürge-Vorsatz, Luisa F Cabeza, Susana Serrano, Camila Barreneche, and Ksenia Petrichenko. 2015. Heating and cooling energy trends and drivers in buildings. *Renewable and Sustainable Energy Reviews* 41 (2015), 85–98.

[27] Jian Qi Wang, Yu Du, and Jing Wang. 2020. LSTM based long-term energy consumption prediction with periodicity. *Energy* 197 (2020), 117197.

[28] Yinjun Wu, Edgar Dobriban, and Susan Davidson. 2020. DeltaGrad: Rapid retraining of machine learning models. In *International Conference on Machine Learning*. PMLR, 10355–10366.

[29] Guohai Xu, Yan Shao, Chenliang Li, Feng-Lin Li, Bin Bi, Ji Zhang, and Haiqing Chen. 2021. AliMe DA: A Data Augmentation Framework for Question Answering in Cold-start Scenarios. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2637–2638.

[30] Lei Xu, Maomao Hu, and Cheng Fan. 2022. Probabilistic electrical load forecasting for buildings using Bayesian deep neural networks. *Journal of Building Engineering* 46 (2022), 103853.

[31] Liang Zhang, Jin Wen, Yanfei Li, Jianli Chen, Yunyang Ye, Yangyang Fu, and William Livingood. 2021. A review of machine learning in building load prediction. *Applied Energy* 285 (2021), 116452.

[32] Yang Zhang, Fuli Feng, Chenxu Wang, Xiangnan He, Meng Wang, Yan Li, and Yongdong Zhang. 2020. How to retrain recommender system? A sequential meta-learning method. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1479–1488.

[33] Fan Zhou and Chengtai Cao. 2021. Overcoming Catastrophic Forgetting in Graph Neural Networks with Experience Replay. *arXiv preprint arXiv:2003.09908* (2021).

[34] Yongchun Zhu, Ruobing Xie, Fuzhen Zhuang, Kaikai Ge, Ying Sun, Xu Zhang, Leyu Lin, and Juan Cao. 2021. Learning to warm up cold item embeddings for cold-start recommendation with meta scaling and shifting networks. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 1167–1176.