2021

PeakTK: An Open Source Toolkit for Peak Forecasting in Energy Systems

Phuthipong Bovornkeeratiroj
University of Massachusetts Amherst

John Wamburu
University of Massachusetts Amherst

David Irwin
University of Massachusetts Amherst

Prashant Shenoy
University of Massachusetts Amherst

PeakTK: An Open Source Toolkit for Peak Forecasting in Energy Systems

Phuthipong Bovornkeeratiroj
University of Massachusetts Amherst
phuthipong@cs.umass.edu

John Wamburu
University of Massachusetts Amherst
jwamburu@cs.umass.edu

David Irwin
University of Massachusetts Amherst
irwin@ecs.umass.edu

Prashant Shenoy
University of Massachusetts Amherst
shenoy@cs.umass.edu

ABSTRACT

As the electric grid undergoes the transition to a carbon free future, many new techniques for optimizing the grid's energy usage and carbon footprint are being designed. A common technique used by many approaches is to reduce the energy usage of the grid's peak demand periods since doing so is beneficial for reducing the carbon usage of the grid. Consequently, the design of peak forecasting methods that predict when and how much peak demand will be seen is at the heart of many energy optimization approaches. In this paper, we present PeakTK, an open-source toolkit and reference datasets for peak forecasting in energy systems. PeakTK implements a range of peak forecasting methods that have been proposed recently and exposes them through well-defined interfaces and library modules. Our goal is to improve reproducibility of energy systems research by providing a common framework for evaluating and comparing new peak forecasting algorithms. Further, PeakTK provides libraries to enable researchers and practitioners to easily incorporate peak forecasting methods into their research when implementing higher level grid optimizations. We discuss the design and implementation of PeakTK and present case studies to demonstrate how PeakTK can be used for forecasting or quantitative comparisons of energy optimization methods.

CCS CONCEPTS

• **Applied computing** → *Forecasting*; • **Hardware** → **Smart grid**.

KEYWORDS

Peak demand prediction, Energy forecasting, Grid optimization

ACM Reference Format:
Phuthipong Bovornkeeratiroj, John Wamburu, David Irwin, and Prashant Shenoy. 2022. PeakTK: An Open Source Toolkit for Peak Forecasting in Energy Systems. In *ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies (COMPASS) (COMPASS '22), June 29-July 1, 2022, Seattle, WA, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3530190.3534791

COMPASS '22, June 29-July 1, 2022, Seattle, WA, USA

ACM ISBN 978-1-4503-9347-8/22/06…$15.00
https://doi.org/10.1145/3530190.3534791

1 INTRODUCTION

The electric grid and associated energy systems are in the midst of an energy transition to a carbon-free future. Decarbonization the world's energy producers and its various consumers is a challenging task and requires incorporating new clean renewable sources into sectors such as buildings, infrastructure, and transportation.

While many techniques have been developed for grid energy optimization, optimizing peak energy use is particularly important from a decarbonization standpoint. For example, peaking power plants that operate during peak periods contribute disproportionately to carbon emission since they tend to be older and less efficient equipment. In recent years, a range of techniques has been proposed for optimizing peak energy use. To illustrate, energy storage batteries have been employed during peak periods to absorb some of the grid's peak demand [24, 29]. Such approaches, called peak shaving, charge the battery during off-peak periods and discharge during peak periods. Similarly, numerous demand-response schemes have been developed to reduce the consumption of electrical loads during peak periods [13, 19, 23]. In addition, flexible EV charging schemes have been developed to be grid friendly by opportunistically charging when grid demand is low and charging less when demand is high [2, 34]. Finally, load scheduling approaches have been designed to shift elastic loads from peak to off-peak periods [5].

A common characteristic of these approaches is that they modulate demand during "peak" periods, e.g., the peak hours of a day or the peak day of the month or season. Consequently, all of these approaches depend on some form of peak prediction to enable this information to be used in the subsequent optimization. For example, battery-based peak shaving methods need an estimate of the *magnitude* of peak demand that will occur so that they shave some of this demand using the battery. Grid-friendly EV charging and load scheduling techniques require an estimate of the times at which peak demand will occur so that they can shift their demand to other off-peak hours. Demand-response schemes need an estimate of which days of the month or season of the year will see peak demand so that demand response can be activated on these days. While the problem of peak prediction or forecasting comes in many flavors, it is an essential building block for a broad range of higher-level grid optimization techniques that improve energy efficiency and enable decarbonization.

A number of peak prediction approaches have been proposed and developed recently, but the researchers face many challenges in this field. First, as in many emerging areas, the field lacks common benchmarks and datasets to enable comparison of various

approaches with newly proposed ones. Second, recent techniques have not made their implementations or datasets available to the community, which impedes reproducibility and requires researchers to re-implement algorithms from prior work in order to compare against them. Re-implementing algorithms from prior work is not an easy task since some details are left to the designer. Finally, the lack of common datasets also implies that experimental results are not directly comparable across papers due to the use of different datasets. As a result, reproducing and assessing those peak forecasting algorithms is challenging.

To address these problems, in this paper, we present *PeakTK*, an open source toolkit that provides reference implementations of a range of peak forecasting techniques and reference energy datasets. Our primary goal is to improve the reproducibility of research results in the field by providing easy-to-use open-source implementations of state-of-the-art peak forecasting approaches, including some of our own, along with reference datasets for experimentation. A related goal is to provide libraries and interfaces to enable researchers and practitioners to incorporate peak forecasting into other research, or commercial systems, then perform higher-level grid optimizations. PeakTK is similar in spirit to other recent open-source energy toolkits such as NILMTK [7] and SolarTK [6] that have provided open-source reference implementations and data for other energy problems and have seen strong adoption by the research community. PeakTK addresses a similar unmet need in the context of energy forecasting, which is a building block for many energy optimization problems. In addition to providing a common framework for evaluating and comparing new peak forecasting algorithms, PeakTK's architecture is extensible—new algorithms and reference datasets can be added to it for use by others, enabling the community to benefit from future research in the area. PeakTK's source code and datasets are available on Github for use by the community: *https://github.com/umassos/peak-tk*.

Implementing a toolkit such as PeakTK poses multiple challenges. First, the ability to collect various peak prediction algorithms in the energy domain and unify them under a common interface that includes a common way to train, run and evaluate algorithms on the same dataset is non-trivial. Second, the ability to reproduce each technique in a manner that faithfully follows the algorithm as described in the original literature is another non-trivial endeavor. Lastly, the ability of the toolkit to run all algorithms in an interoperable manner that enables direct comparison with each other is also non-trivial and poses a great challenge to reproduce. All these items present great challenges that must be overcome both from a research and implementation perspective, and demonstrate the importance of making such a tool available to the community.

In designing and implementing PeakTK, we make the following contributions;

(1) **Peak Forecasting Taxonomy**. We provide a taxonomy of peak forecasting and modeling problems in the literature and discuss various state of the art methods. Using this taxonomy, we discuss several state of the art techniques for determining both the magnitude of the peak demand and when it occurs.
(2) **PeakTK Reference Implementations**. We present the extensible architecture of the PeakTK framework and its interfaces exposed to users and applications for generating energy forecasting. We then discuss reference implementations of several state-of-the-art forecasting methods for peak forecasting problems in PeakTK, i.e., peak demand prediction, peak hour, peak day of the month, and peak day of the year prediction and also describe the energy datasets included in the toolkit for experimental comparisons.
(3) **Case Studies**. Finally, we present multiple case studies to demonstrate how PeakTK can be used for forecasting and quantitative comparison of energy optimization methods. Our case studies show how PeakTK can be incorporated into battery scheduling techniques, as well as how PeakTK can be used to compare the performance of different peak prediction approaches using reference datasets.

2 BACKGROUND

In this section, we provide a taxonomy of peak forecasting techniques and discuss several state of the art techniques that have been proposed recently.

Peak forecasting can be viewed as the problem of estimating, or predicting, the actual peak energy demand over a certain time window as well as the duration within that window when the peak demand will occur. Thus, there are two components to peak forecasting: (i) determining the *magnitude* of the energy demand, and (ii) determining the *time* when the peak energy demand occurs. The *time window* over which both of these factors are estimated also yields different flavors to the forecasting problem. While some higher-level techniques require estimates of both the magnitude of the peak and time duration when it occurs, other techniques only need a prediction of one of these components. For example, an intelligent EV charging algorithm may only need a prediction of when the peak grid demand will occur, so that it can defer EV charging to other times. As a result, the actual magnitude of the peak is not important in this case. In contrast, a peak shaving technique using batteries may need predictions of the size of the peak and the duration when it occurs, so that it can determine how much of that peak to shave and when to operate the batteries to do so.

The literature has studied both components of peak forecasting, together and separately, and also studied the problem over different time granularities, e.g., a day, month, year, etc. Figure 1 shows a taxonomy of various approaches.

2.1 Determining When the Peak Occurs

Many peak forecasting techniques operate on a daily basis, where they estimate the peak hours of the day (usually in the evening) when the peak demand will occur. We call this peak hour forecasting, and it is useful for a range of methods, including battery-based peak shaving, EV charging, and more. Other techniques operate over a monthly or yearly time window. The peak day of the month problem involves determining the one day within each month that will see the greatest energy demand. This technique is useful for estimating (and reducing) demand charges, or the so-called peak surcharge, where a consumer pays a surcharge on their monthly bill based on their consumption during the peak day in each month; lowering consumption in such periods yields a reduction in monthly energy bills. The peak day of the year (or season) problem involves

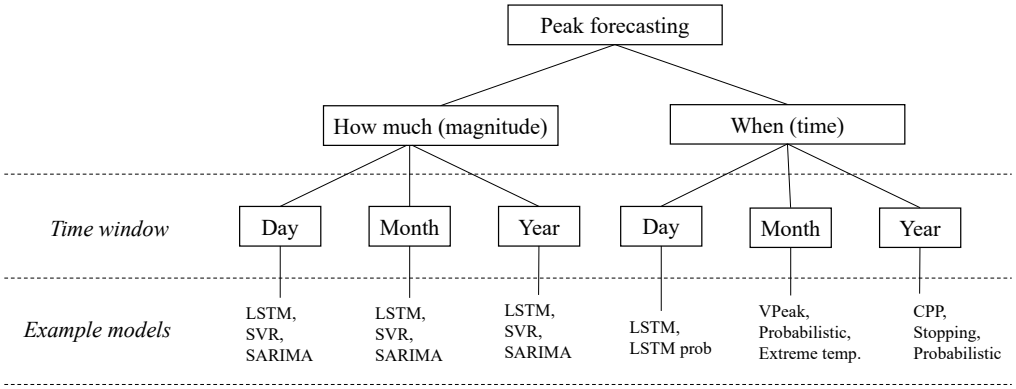


Figure 1: Taxonomy of various peak prediction approaches.

determining the peak day, or the top-N peak days, during each year. These peaks days are often used by utilities and grid operators to set annual capacity charges for customers. Regardless of whether the time window is a day, a month, or a year, the forecasting problem involves estimating *when* the peak demand will occur and not the magnitude of the demand.

2.2 Determining the Magnitude of the Peak Demand

The other component of peak forecasting is to determine the magnitude of the peak demand, which can either be the mean value or a time series of energy demand over the peak demand window. This problem is also related to the well-studied problem of demand forecasting or load forecasting, which involves determining the future energy demand over some time window. Over a time window of a day, peak demand forecasting may involve predicting the hourly demand during the peak period, which is often in the evening. Over a monthly or yearly granularity, it may involve predicting the daily energy demand on the peak day of the month or the peak day of the year. One approach for peak demand forecasting is to first perform normal demand forecasting to derive a time series of demand over the whole window and then simply choose the highest predicted values as the peak demand. An alternative approach is to directly predict the peak demand values, using, for instance, the historical peak values.

2.3 Types of Peak Prediction Algorithms

Having discussed the taxonomy, we now present an overview of the approaches that have been proposed in the literature for each type of prediction. Our goal here is to discuss various approaches using the above taxonomy, while deferring a detailed discussion and implementation of these algorithms to Section 3.2.

Peak days of the year prediction. The goal of this approach is to predict the n days in a year that have the highest peak demand. The results of this prediction are used by utility companies to assess a surcharge to their customers, specifically large commercial and industrial customers for their consumption during these peak days. For example, in the 5 Coincident Peaks (5CP) program in Ontario, Canada, utility companies charge their commercial customers for their contribution to the load during the peak days of the year [18]. This also allows utility companies to set aside additional energy resources in preparation for the expected higher demand during these peak days. Similarly, consumers themselves can use the results of such predictions to predict the expected peak days and curtail their load to avoid the surcharges.

Peak days of the month prediction. In this approach, the goal is to predict the n days in a month with the highest energy demand. Similar to the previous approach, the results of this prediction can be used by utility companies to determine a peak surcharge for their large customers, as well as customers using such results to know when to curtail their load in order to avoid extra peak charges. A number of grid energy optimizations can also be implemented using this approach. For instance, VPeak [8] uses a prediction of the peak days of the month to devise an optimal policy of utilizing volunteered resources for peak shaving.

Peak hours of the day prediction. This involves predicting the hour of the day during which the grid is expected to see the highest energy demand. To determine the expected peak hour, the energy demand for the next day is first predicted, and the hour with the highest demand is selected as the peak hour. Using this approach, a variety of grid energy optimizations such as battery-driven peak shaving can be implemented. For example, Soman et al. [31] propose a deep learning based approach that predicts the k hours of each day with the highest and lowest demand for micro-grids. Using this peak forecasting technique, a battery-based peak shaving approach is implemented for a micro-grid resulting in significant annual energy savings. Similarly, Liu and Brown [22] propose an LSTM model that predicts the peak demand 24 hours ahead which not only helps to improve the reliability of electricity supply, but also helps consumers to avoid surcharges that are charged for electricity consumption during peak periods.

Peak demand prediction. The goal of this approach is to provide the estimated peak demand within a specified time frame. Utility companies can use this approach to plan ahead for expected peak demand, e.g., by incorporating a specific amount of energy storage into the grid to absorb the extra energy demand. Peak demand prediction uses historical energy and weather data along

with weather forecast data to predict expected demand in the future. Various approaches utilizing time series analysis and machine learning methods have been proposed to solve this problem [3, 8, 9, 18, 22, 31].

Table 1 shows a summary of the existing approaches. For each approach, the table shows the capabilities and the various types of predictions the algorithm supports. Various datasets have also been introduced to evaluate the performance of each of these approaches in peak prediction. PeakTK aims to unify these approaches into a single toolkit that exposes a well-defined interface. PeakTK also allows customization of each approach based on the parameters of each algorithm. In addition, PeakTK allows for the implementation of custom and new algorithms using the same unified interface. We also provide reference datasets that can be used to evaluate the performance of new algorithms compared to preexisting approaches.

2.4 Related Energy Toolkits

To the best of our knowledge, there is currently no open-source or closed-source toolkit that is specifically designed for peak forecasting in energy systems. However, there are other energy toolkits as well as more general-purpose toolkits that are related to electricity demand prediction, inferring energy usage, and rare-event detection, which we discuss below.

PeakTK is directly inspired by two prior open-source toolkits, NILMTK and SokarTK, that have provided reference implementations of energy algorithms and datasets for other energy problems. Due to the open-source nature of both toolkits, they have seen good adoption by the research community and enabled comparable and reproducible experimental results in many subsequent publications. We designed PeakTK with a similar goal in mind.

NILMTK [7] is a well-known open-source toolkit in the energy domain that provides source implementations of a range of non-intrusive load monitoring (NILM) algorithms such as Combinatorial Optimization and Factorial Hidden Markov Models. A NILM algorithm breaks down ("disaggregates") a household's aggregate electricity demand into individual load-level components, which can then be used to analyze the energy usage of the household. The toolkit also includes public data sets, standard data structure, statistics, diagnostic functions, and accuracy metrics for benchmarking NILM algorithms. NILMTK uses an extensible design, and other researchers have contributed to its repository of algorithm implementations and datasets, a feature that we adopt for PeakTK as well. SolarTK [6] is another open-source toolkit for modeling and forecasting the output of residential solar arrays. SolarTK implements various forecasting algorithms that take the location, weather, and historical data as inputs to learn a model that can predict the future solar generation from that array. PeakTK's models can estimate the physical specifications, the maximum generation potential, and the weather-adjusted generation of a solar site. The toolkit also includes a microgrid dataset of hundreds of households, including solar power generation and energy consumption. Both NILMTK and SolarTK are open source and implemented in Python.

Time-series forecasting is a general forecasting technique that can predict future values of a time series based on the given historical trends. However, they are not specifically built for the energy domain and substantial effort is needed on the part of a researcher to develop and train a time series model for energy or peak forecasting. PeakTK seeks to simplify this task by providing pre-built time series-based forecasting methods for peak energy prediction from recent literature.

More broadly, Prophet [32] is an open-source software for time-series data forecasting developed by Facebook. It supports seasonality, such as seasonal data or daily/weekly/monthly/yearly patterns. The software is implemented in R and Python. FABLE [25] is also an open-source collection of commonly used time series forecasting models such as the ARIMA model, Exponential smoothing state-space model, time series linear models, simple method for benchmarking, and neural network autoregressors. The framework also provides the tools to evaluate and visualize.

Another type of tool related to peak prediction is anomaly detection or extreme event detection, which focuses on detecting an anomaly in time-series data. For example, luminol [21] is a python library for time-series data analysis which has anomaly detection as its main feature. The library is developed by LinkedIn. Python toolkit for detecting outlying objects (PyOD) [35] is a toolkit that has more than 30 detection algorithms and has been widely used in academic research and commercial products. Telemanom [17] is an anomaly detection using LSTMs and automatic thresholding. The tool is implemented in Python using Keras/Tensorflow. While peak detection can be viewed broadly as a type of anomaly detection, adopting such techniques to the energy domain requires careful design and training, while PeakTK provides many pre-built reference implementations to simplify this task for researchers.

3 PEAKTK PEAK ENERGY FORECASTING TOOLKIT

In this section, we discuss PeakTK's architecture, components and implementation. We also define PeakTK's interfaces and provide example uses in the latter part of this section.

3.1 PeakTK Design and Architecture

At a high level, PeakTK is a collection of data-driven algorithms and machine learning techniques that perform two main functions: (i) *learn* an energy model and its parameters, and (ii) *predict* the peak energy demand using the learned model, both in terms of the timing and magnitude of the peak. From a user's perspective, PeakTK exposes a set of interfaces that enable a user to perform these functions while abstracting the lower-level implementation details of each approach from the user.

The primary goal of PeakTK is to provide reference implementations and datasets to enable a common framework for experimentation, quantitative comparisons, and use in higher-level energy optimizations. To do so, PeakTK's design strives for simplicity and provides extensibility. The interfaces provided by the toolkit are easy to use and understand, making PeakTK user-friendly for the end-user. All features are encapsulated into their respective modules. The interfaces only expose necessary and essential functions while hiding unnecessary details from the user. The second goal is to make the toolkit extensible by allowing developers to add new data and algorithms to the toolkit easily. We provide a list of basic API functions that new objects have to implement. This list serves as a guide for developers and ensures that new objects are

Table 1: An Overview of the Algorithms in PeakTK toolkit

Algorithm	Type	Type of Prediction
LSTM hourly demand prediction [31]	Machine learning	Next day hourly demand prediction
LSTM probabilistic classification [22]	Machine learning	Next day peak hour prediction
VPeak [8]	Machine learning	Peak days of the month prediction
Probabilistic peak day prediction [18]	Time series analysis	Peak days of the month prediction
Extreme Temperature approach	Time series analysis	Peak days of the month prediction
CPP [14]	Time series analysis	Peak days of the year prediction
Stopping approach [3]	Time series analysis	Peak days of the year prediction
Probabilistic peak prediction [18]	Time series analysis	Peak days of the year prediction
SARIMA	Time series analysis	Demand prediction
SVR	Machine learning	Demand prediction
LSTM-based demand forecasting [9]	Machine learning	Demand prediction

compatible with existing modules. Moreover, to keep the interface simple and compact, we make sure that the number of methods a new class must implement is minimal. The toolkit is implemented in Python 3 which is known for being powerful, fast, and easy to learn. The language has been widely used by many Machine Learning libraries such as Sklearn [28], Tensorflow [1], and Pytorch [27]. This allows PeakTK to access and utilize these libraries, since machine learning and data-driven approaches are at the core of peak prediction techniques.

Figure 2 depicts an overview of PeakTK. As shown in the figure, model learning and prediction are the two main logical components of the toolkit. In the *learning* phase, PeakTK adopts a data-driven approach to manipulate and learn an energy model using input data. During learning, PeakTK can use either a machine learning or a time-series analysis based approach depending on the specific algorithm chosen by the user. Some aspects of peak prediction also make use of exogenous factors such as weather data to perform prediction, i.e., by learning the relationship between energy usage and weather parameters, such as temperature, the expected demand can be predicted by analyzing the weather forecast. In such cases, weather forecast data can be supplied as input during prediction to enable such prediction. Finally, in the *prediction* phase, the learned model is used to generate predictions, i.e., peak hours of the day, peak days of the month, peak days of the year, or the expected peak demand.

Depending on whether the learned energy model is machine learning or time series based, different modules are loaded to support the specified approach. We now discuss some of these functions and the modules that implement them in detail.

Preprocessing. This functionality is provided by the data preprocessor module which implements various pre-processing functions that are commonly used for preparing the input data for peak prediction models such as loading, peak day labeling, upsampling and downsampling, data splitting, and data scaling.

Loading. This functionality is provided by the labeling component which is responsible for converting the raw input data into PeakTK's data input format, which is used throughout the pipeline. Developers can easily develop an importer for a new dataset by implementing the importer interface. PeakTK's importer generally requires historical electric load data and weather data. The toolkit also provides a helper function for merging two datasets as long as they have a common index, e.g., date time. Currently, PeakTK provides importers for ISO-NE[1], ESO[2], and Smart* apartment data[3] and weather data from the DarkSky API[4] (See Table 3).

Labeling. Data labeling is provided by the labeling helper component which assigns a label to the n peak days of each month or year based on the user goal. n can be varied based on the number of peak days the user wishes to predict. This is especially useful for machine learning based approaches which rely on the labeled data to train, validate and test the learned model.

Splitting and scaling. For machine learning based approaches, PeakTK provides time-based splitting by allowing the user to specify the date and time or percentage of the data in each split using the data splitter component. We also provide data scaling functions using the Min-Max Scaler, Standard Scaler and Max-Abs Scaler from the Sklearn library [28].

3.2 Peak Forecasting Algorithms and Reference Implementations

PeatTK implements a variety of peak and demand prediction algorithms. In this section, we discuss various algorithms for peak energy prediction and their reference implementations in PeakTK.

3.2.1 Peak hours of the day prediction. Two peak hour of the day prediction algorithms are supported in PeakTK. Both of these algorithms are based on the popular LSTM deep learning architecture.

(1) **LSTM-based hourly probabilistic classification.** Proposed by Liu and Brown [22], this approach uses an LSTM model to generate next day hourly probabilities for how likely it is for a particular hour of the day to be a peak hour. LSTM is a variant of Recurrent Neural Network (RNN) which is known for working well at capturing the sequential dependencies in time-series data since it learns from the previous state information to make prediction decisions. In PeakTK, we use this LSTM model to generate all next day hourly peak probabilities (24 hours) and then use descending sort order to determine the top n peak hours of the day.
The proposed model architecture is as follows. First, the model is made up of 6 hidden layers, each containing 24 neurons. Next, the authors set the look back distance to 24 — since the model operates at hourly granularity, this distance enables the model to learn from the past 24 hours

[1] https://www.iso-ne.com/isoexpress/web/reports/load-and-demand
[2] https://data.nationalgrideso.com/data-groups/demand
[3] http://traces.cs.umass.edu/index.php/Smart/Smart
[4] https://darksky.net/dev

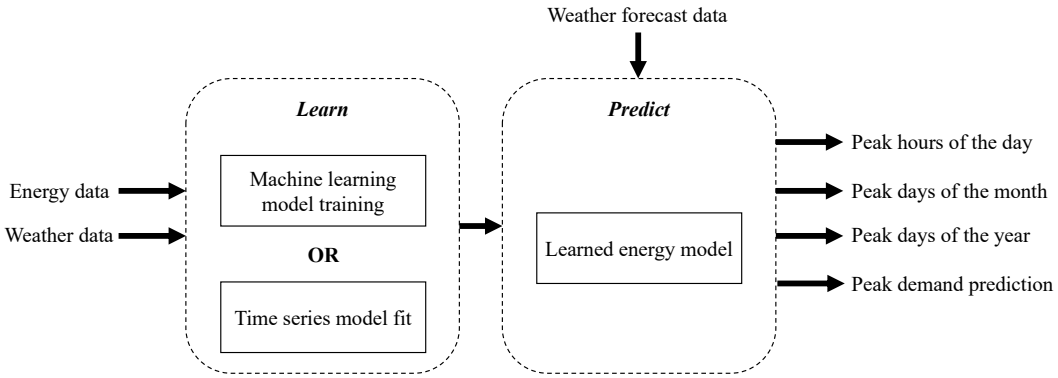


Figure 2: Overview of PeakTK.

i.e., one day. Note that the ratio of peak to non-peak hours is highly imbalanced i.e., out of 24 hours in a day, the goal is to predict the one peak hour — a 1:23 ratio. This may lead to disproportionately low probabilities for the positive class. To address this problem, the authors apply Equation 1 to re-compute the final probabilities, where P represents the probability generated by the model. Since summer and winter seasons present different usage patterns, the authors propose building two separate models for the two seasons.

$$P_{final} = 0.1 \times \sqrt{P_{model} \times 100} \quad (1)$$

To evaluate the performance of the proposed model, the authors use a power demand dataset collected from the city of Ontario, Canada. The data contains power demand recorded over a 5-year period (2003-2008), and is recorded in hourly granularity. This model outperforms all other approaches evaluated by the authors and achieves 0.68, 0.87, and 0.98 in precision, recall and accuracy scores respectively for winter predictions. For summer predictions, this LSTM model achieves 0.42, 0.44, and 0.95 in precision, recall, and accuracy scores respectively. PeakTK implements this model as is, and performs the distinction between seasons under the hood, thereby abstracting all implementation details from the end user.

(2) **LSTM-based hourly demand prediction.** Next, we adopt the LSTM-based hourly demand prediction approach proposed by Soman et al. [31] for peak hour prediction. The authors propose an LSTM-based peak forecasting approach that is designed for predicting the $top - k$ and $bottom - k$ high and low demand hours for a day. Using the output of this model, PeakTK selects the top n peak hours of a day depending on the number of hours for which prediction is required. The model requires historic hourly demand over a 2-day period to make predictions over the next 24 hours i.e., top-k hours of the day that experience the highest demand. The authors implement two model variants. First, a 2-layer configuration that is made up of 100 and 80 neurons in each layer, and second, a 4-layer configuration that is made up of 100, 90, 80, and 70 neurons in each hidden layer respectively. To train the model, the authors use Adam optimizer with an adaptive learning rate of 0.1 to 0.005, and 0.2 drop out. To select optimal hyperparameters, the authors use grid search for both model configurations. The authors then evaluate the performance of the proposed models using a real world dataset collected from a micro-grid of 156 buildings. The 4-layer model outperforms all other prediction models in all evaluations ranging from 1-5 peak hours of the day predictions. In PeakTK, we adopt the 4-layer model as the second peak hour of the day prediction algorithm. We note that this approach is extensible, and can be used to predict more than 1 peak hour of the day by varying the value of n. Our implementation comes configured with default hyperparameters (as specified in the original work), with the ability to specify different values for different peak prediction environments.

3.2.2 Peak days of the month prediction. PeakTK provides an implementation of three peak days of the month approaches.

(1) **Probabilistic approach.** PeakTK provides a modified version of the original probabilistic approach, which aims to pick peak days of the year, proposed by Jiang et al. [18]. The approach chooses a day with predicted demand and forecast temperature higher than the peak demand and extreme temperature thresholds, then calculates the probability that it will be a peak day by using 14-day demand forecast and the peak day it has seen so far. More details on how the algorithm originally works can be found in 3.2.3. In the peak day of the month version, we add a separate peak demand threshold and extreme temperature threshold for each month since each month can have a different load profile and climate. The extreme temperature can also be ignored in some months. For instance, the temperatures in spring and fall are usually mild and consistent over the month. Hence, using an extreme temperature threshold does not help select the peak day of those months. The rest of the approach works as in the case of yearly peak days prediction with up to 14 days of demand forecast look ahead.

(2) **Extreme temperature approach**. Temperature is one of the main primary factors that can help determine the magnitude of electric demand. When the temperature is high, air

conditioners will be turned on. On the other hand, the temperature is low. The difference between room temperature and outside temperature also has an impact on how much energy is used for cooling or heating. The extreme temperature approach utilizes these facts to determine whether tomorrow is a peak day or not using just the predicted temperature. By giving historical temperature data, quantile value, and summer/winter months, the model learns the extreme temperature value as a threshold for each month. And to select a peak day, the model compares the forecast temperature with the threshold of that month. If the forecast temperature is considered extreme (lower than the threshold value for winter or higher for summer), that day will be predicted as a peak day.

(3) **VPeak.** VPeak [8] is a machine learning and threshold-based approach for predicting the peak days of the month. The algorithm computes the cumulative distribution function (CDF) of the peak days distributed over each month in past years and uses that distribution to select peak days in the future. The threshold is adjusted by comparing the ratio of selected peak days with that of the historical distribution. If the ratio is lower than that in the distribution, the algorithm becomes more aggressive in picking peak days, i.e., by reducing the peak threshold, and thus more days are selected as peak ones due to the lower threshold. On the other hand, the algorithm becomes more conservative in selecting peak days if the number of peak days selected for the month is higher than the expected value from the historical distribution. Therefore, the peak threshold is adjusted upwards, which results in fewer peak days being selected due to the high peak threshold VPeak requires for the main inputs. First, historical energy data is required for computing the historical CDF. Second, the peak demand threshold for each month in the historical data is also required. VPeak uses this information to adjust the threshold based on the season of prediction. Third, the threshold update frequency which determines how often the peak threshold should be updated is also required as input. For example, if the frequency is 3, the threshold is updated every three days. Finally, the number of peak days to select per month is also required as input. Using this information, VPeak outputs the top n peak days of a month.

3.2.3 Peak days of the year prediction. For this problem, three peak days of the year prediction techniques are implemented.

(1) **CPP approach.** The CPP approach is a threshold-based peak day prediction algorithm used by a California utility [14]. CPP requires demand forecasting to predict the absolute peak demand expected on a particular day. The peak demand is then compared to the peak threshold, which can be adjusted dynamically over time, to determine whether a particular day is a peak day or not. If the peak demand on a day is higher than the threshold, then the day is labeled as a peak day. Conversely, days whose peak demand falls below the peak threshold are considered non-peak days. In PeakTK, the CPP approach requires 4 main parameters. First, the number of peak days n, which should be selected as peak days for the period. Second, the initial peak threshold T is also required which represents a starting point for peak day selection. Over time, this threshold is adjusted by CPP based on observed conditions. Third, CPP also requires a threshold adjustment value which is either added or subtracted from the threshold depending on the number of peak days selected by CPP at a particular time. Finally, a list of the days in a month during which the threshold should be adjusted can also be provided. CPP uses this list to determine the days of a month during which the threshold can be adjusted. For example, if 1 and 15 are provided, CPP adjusts the peak threshold every 1st and 15th day of the month.

(2) **Stopping approach.** Optimal stopping is a problem of choosing the time to take an action that is likely to maximize the expected reward or minimize the expected cost. For example, the secretary problem involves choosing the best secretary from the pool but with some special rules. First, the interviewer will interview and gain information about one secretary at a time and must decide immediately after the interview whether to hire or reject that secretary. The information about the subsequent applicants is unknown, and rejected secretary cannot be recalled. The goal of the problem is to hire the best secretary. There are other similar problems, such as the marriage problem or the sultan's dowry problem. Jiang et al. [18] proposed applying the idea of the matroid secretary problem, which was introduced by Babaioff, Immorlica, and Kleinberg [3], to the peak day prediction problem since picking k-peakest day in the given year is similar to the k-secretary problem. The basic form of the stopping approach in PeakTK is as the following: first, this approach first observes m number of days of the year and selects the highest day demand as the peak threshold. Then, from the $(m+1)^{th}$ day onwards, any day with predicted demand exceeding the peak threshold is considered a peak day, while any day with predicted demand that falls below the threshold is marked as a non-peak day. In PeakTK, this algorithm requires as input the number of days to observe m and the number of peak days to predict n. The algorithm stops when n peak days have been picked, or it reaches the end of the year.

(3) **Probabilistic approach.** Proposed by Jiang et al. [18], the key idea of this technique is to compute the probability that the next day will be one of the n peak days based on the peak days it has seen so far, and the look-ahead forecast days it can see. The algorithm applies a look ahead approach which analyzes up to 14-day energy demand forecast to generate predictions. In the original implementation, 14-day short-term demand forecast was used as the look ahead period. In PeakTK, this parameter can be tuned to support the number of available demand forecast days since 14-day energy demand forecasts may not always be available. In addition to the number of look ahead forecast days, the implementation of this algorithm also supports the peak demand threshold, peak probability threshold, extreme temperature threshold and the number of peak days n as parameters. To determine whether a day is a peak day or not, the algorithm compares

the day's predicted demand with the peak threshold as well as the temperature with the temperature threshold. These two thresholds are also updated over time for more accurate prediction. Lastly, it is important to note that the look ahead demand forecast should be provided as additional input to this algorithm.

3.2.4 Demand prediction. PeakTK provides the implementation of three demand forecasting algorithms.

(1) **SARIMA.** Seasonal ARIMA is the extension of ARIMA model, a well-known time-series forecasting which combines the differencing with AutoRegressive (AR) and Moving Average (MA). SARIMA extends ARIMA by adding a capability to model seasonal data. It has been widely used in many areas such as tourism forecasting [16], traffic flow forecasting [10], and energy demand prediction [11] which are heavily influenced by seasonal weather changes. PeakTK's SARIMA implementation uses the Statsmodel library [30] and supports the order (p, d, q) and seasonal order (P, D, Q) parameters only. This is because these are the most relevant to demand and peak day prediction. The remaining parameters use the default values as defined in the library. Finally, the number of observations per year m is also provided as a parameter.

(2) **SVR-based demand forecasting.** Support Vector Regressor (SVR), a regressor version of the Support Vector Machine (SVM) is a supervised machine learning technique that can predict discrete values. SVR has been used widely for predicting electric load demand [12, 33]. The goal of SVR is to find the best hyperplane that can accurately estimate the output values from the input. SVR allows user to define an acceptable error margin ϵ and the regularization value C which can increase/decrease the penalization from the slack values ξ. The SVR find the optimal hyperplane by minimizing the following function:

$$min\frac{1}{2}||W||^2 + C\sum_{i=1}^{N}(\xi_i + \xi_i*) \tag{2}$$

Subject to the following constraints:

$$\begin{cases} y_i - (w, \phi, (x_i)) - b \leq \epsilon + \xi_i \\ (w, \phi, (x_i)) + b - y_i \leq \epsilon + \xi_i* \\ \xi_i, \xi_i* \geq 0 \end{cases} \tag{3}$$

PeakTK's implementation of SVR uses the Sklearn library [28].

(3) **LSTM-based demand forecasting.** Lastly, PeakTK provides a deep learning based demand forecasting approach that is based on PowerLSTM proposed by Cheng et al. [9]. PowerLSTM is made up of a two-layered LSTM network consisting of 32 and 16 neurons each respectively. To train and evaluate the model, the authors use a real world and publicly available power usage dataset [4]. The authors use one month of data to train and evaluate the model — training is performed on the first 28 days of the month of July, while prediction is performed on the remaining three days of the month. The authors show that PowerLSTM significantly outperforms other demand prediction approaches by up to 28.58% in MSE reduction. Since PowerLSTM is a relatively simple architecture, PeakTK provides its implementation as one of the demand forecasting algorithms. The implementation in PeakTK requires 7 days of historical demand, 7 days of weather information, as well as the next day's weather forecast to predict the next day's peak demand.

3.3 PeakTK Implementation

PeakTK is implemented in Python3. We also provide a packaged installer that can be installed via *pip*. As noted earlier, our implementation relies on several Python libraries, such as the Pandas data analysis library [26], NumPy scientific computing python library [15], Sklearn machine learning and predictive data analysis library [28], and the Statsmodel statistical modeling library [30]. We also use imbalanced-learn [20], a library for re-sampling imbalanced datasets, and Tensorflow [27], a deep learning library. We expose a modularized API for performing various prediction tasks. We also provide numerous examples of using the library in the documentation section.

For instance, Figure 4 shows the process of using PeakTK to perform peak days of the month prediction on one of the reference datasets. The example process covers all essential steps, including loading the dataset into PeakTK's data input format, preprocessing and splitting the data for training and testing, using demand prediction and peak prediction algorithms, and evaluating the results. Each step is encapsulated into one or a few lines of code which is convenient for the user to make changes. For example, if the user wants to test an algorithm on another dataset, they can simply change one line of code to load a different dataset and run the pipeline again. Switching algorithms also requires changing just a few lines of code.

Table 2 shows a summary of all modules available in PeakTK. The four main modules i.e. *demand_prediction, peakday_prediction_yearly, peakday_prediction_monthly, and peakhour_prediction*, contain the implementation of peak prediction algorithms discussed in 3.2. The *preprocessing module* includes helper functions for manipulating the data such as scaling, train-test splitting and oversampling/undersampling. The *dataloader module* provides code for converting the reference datasets into PeakTK's data input format. The *stats subpackage* contains statistic-related helper functions such as ground truth peak day labeling. Lastly, the *metrics module* includes accuracy metric functions (e.g., recall, precision) for evaluating the results.

3.4 Reference datasets

Along with the PeakTK library, we include three reference datasets i.e., two grid datasets from different geographic locations as well as different climatic conditions, and a microgrid (400+ houses) electric usage dataset. All datasets are publicly available and can be downloaded directly from the official sources indicated for each dataset. In our case, each reference dataset is a subset of the original dataset that is curated and cleaned for ingestion by PeakTK's forecasting algorithms.

While the included datasets share multiple similarities i.e., they are all grid electric usage datasets, they also possess structural differences that make them suitable for varying prediction tasks. First,

Table 2: Summary of PeakTK modules

PeakTK package	Description
peaktk.demand_prediction subpackage	peak demand prediction routines
- peaktk.demand_prediction.SARIMA	
- peaktk.demand_prediction.SVR	
- peaktk.demand_prediction.LSTM_DP	
peaktk.peakday_prediction_yearly subpackage	peak day of the year prediction routines
- peaktk.peakday_prediction_yearly.CPP_approach	
- peaktk.peakday_prediction_yearly.stopping_approach	
- peaktk.peakday_prediction_yearly.probabilistic_yearly	
peaktk.peakday_prediction_monthly subpackage	peak day of the month prediction routines
- peaktk.peakday_prediction_monthly.VPeak	
- peaktk.peakday_prediction_monthly.probabilistic_monthly	
- peaktk.peakday_prediction_monthly.extreme_temperature	
peaktk.peakhour_prediction subpackage	peak hour of the day prediction routines
- peaktk.peakhour_prediction.LSTM_peakhour	
- peaktk.peakhour_prediction.LSTM_probabilistic	
peaktk.preprocessing subpackage	data preprocessing routines
- peaktk.preprocessing.helper	
- peaktk.preprocessing.imbalance	
- peaktk.preprocessing.preprocessing	
peaktk.dataloader subpackage	dataset helper routines
- peaktk.dataloader.ESO_helper	
- peaktk.dataloader.ISONE_helper	
- peaktk.dataloader.SmartStar_helper	
peaktk.stats subpackage	statistics computational routines
- peaktk.stats.peakday_helper	
- peaktk.stats.threshold_helper	
peaktk.metrics subpackage	evaluation metric routines
- peaktk.metrics.metrics	

the included datasets have been collected from different geographical locations that experience different climatic conditions. This helps in evaluating the generalizability of the algorithms implemented in PeakTK by performing prediction tasks on data with varying seasonal patterns. For instance, data collected from the North Eastern region of the United States portrays different summer and winter intensities from European sourced data. Second, the datasets are available at varying granularities i.e., data is recorded at hourly, 30-minute and 15-minute intervals. This enables the comparison of peak prediction algorithms that perform prediction at fine and coarse granularities. The datasets also have different amounts of demand forecast data, with one of the datasets lacking any form of look ahead information. This enables PeakTK to evaluate algorithms that require look ahead information, as well as those that do not. Finally, the datasets are available across different time durations. This heterogeneity of the datasets included in PeakTK enables the implementation and evaluation of a wide range of peak prediction algorithms and covers most use cases in peak prediction tasks.

ISO New England dataset. The ISO New England dataset [5] contains electricity consumption data in the New England region of the USA. The data consists of three-day energy demand forecast at hourly granularity and historical electricity demand at 5-minute granularity. This makes ISO an ideal dataset for peak prediction and evaluation tasks. In PeakTK, we include ISO-NE hourly demand data from the three-year period from 2018 to 2020. Figure 3a depicts the energy demand along with temperature data for the New England region across the whole period. Energy demand depicts a yearly bi-modal peak i.e., during December-February and June-August. This coincides with winter and summer seasons in this region. The figure also shows that temperature has an inverse relationship with energy demand during winter months i.e., as the temperature decreases, the energy demand increases, and a direct relationship during summer months i.e., as the temperature rises during summer, the aggregate energy demand also increases. This can be attributed to heating and cooling requirements respectively.

NationalGrid ESO dataset. The National Grid releases open data from Great Britain's energy usage operators. [6] The data contains up to 14-day ahead demand forecast as well as historical demand information at half-hour granularity. In PeakTK, we include 30-minute ESO historical demand for the 5 year period from 2016 to 2020. Figure 3b depicts the energy demand in the region across the whole period. Energy demand indicates a yearly unimodal peak i.e., between January-March, which coincides with winter months in the region. The figure also shows that temperature has an inverse relationship with energy demand i.e., as temperature decreases during winter months, the energy demand increases due to increased heating requirements. As the temperature increases during the summer months, the energy demand does not increase linearly with temperature, indicating that cooling demand in the region is lower than heating energy demand.

Smart* dataset. The Smart* dataset[7] contains minute-level electricity usage data from 400+ anonymous homes. Due to the disaggregated nature of the dataset, peak prediction can be performed at micro-grid level e.g. for a collection of homes using this dataset. PeakTK includes the Smart* Apartment data released in 2017 from the one-year period between 2015 and 2016. The apartment dataset contains data collected from 114 single-family apartments. Figure

[5] https://www.iso-ne.com/isoexpress/web/reports/load-and-demand

[6] https://data.nationalgrideso.com/data-groups/demand

[7] http://traces.cs.umass.edu/index.php/Smart/Smart

3c depicts the aggregate demand from all apartments in the dataset, along with the temperature data during the same period. Energy demand shows a unimodal peak, with the electric demand being lowest during summer months, and highest during winter months. We attribute this observation to the fact that the apartment homes from which this data was collected lack HVAC units, and therefore, the overall energy demand during summer is made up of internal appliances only. However, space heating in the apartments uses electric energy, hence the high energy demand during winter months.

Weather data. In addition to energy forecast and historical demand data, we also provide weather data which is required by some algorithms for peak prediction. For each of the three reference datasets, we select a representative location for the region and gather weather data at hourly granularity from the DarkSky API [8]. Since Dark Sky is scheduled to be deprecated at the end of 2022, PeakTK also plans to support other weather data sources such as Tomorrow.io Weather API [9].

4 PEAKTK CASE STUDIES

In this section, we illustrate several use cases for the PeakTK toolkit. Since the algorithms implemented in PeakTK were published elsewhere, the goal of our experimental case study is not to perform detailed comparisons — rather, we illustrate how PeakTK's modular approach simplifies comparisons between forecasting methods. We also show how PeakTK can be incorporated into higher-level energy optimization techniques such as battery scheduling.

4.1 Peak Demand Forecasting

Our first case study shows how PeakTK can be employed to estimate the magnitude of the peak demand in the future. For this case study, we choose the time window to be one day and use PeakTK's LSTM-based demand forecasting algorithm. We train the LSTM model using the ISO New England dataset and weather data for Boston, USA. We then use test data from the ISO dataset to make predictions using the trained LSTM model. We test the model over a five month test period (June to October).

Figure 5a depicts the result of this experiment. The figure shows that the LSTM model is able to predict energy demand accurately, including months of high energy usage e.g., August, as well as months of lower energy usage e.g., October. To measure the quality of prediction, we compute the RMSE between the actual and predicted demand for each day in the test period. The RMSE is 1358.78, which corresponds with prediction results made in the original study. Similarly, Figure 5b depicts the actual and predicted demand over a one-week period. The figure shows the variation in demand across different days of the week e.g., weekend usage is lower than weekday usage. The figure also shows that the LSTM model performs well on this prediction task with an RMSE of 1091.49, which corresponds to the results observed in the original study.

4.2 Peak Hours and Day Forecasting

Our second case study illustrates how PeakTK can be used to forecast the specific hours in a day or days in a month or year during which peak demand is experienced. A secondary goal of this experiment is to demonstrate the comparability of various algorithms on the same prediction task. Using the ISO New England dataset for the year 2020, we use PeakTK to predict peak hours using different algorithms, thereby showing how PeakTK can be used to compare the performance of various algorithms on the same prediction task.

We begin our analysis by examining the performance of two peak hour of the day prediction algorithms implemented in PeakTK. Figure 6a depicts the performance of two peak hour prediction algorithms i.e., LSTM and LSTM probabilistic on the same dataset. The figure shows that the LSTM model outperforms the probabilistic variants on all the metrics evaluated. LSTM probabilistic achieves precision, recall, and balanced accuracy scores of 0.78, 0.78, and 0.83, respectively. On the other hand, the LSTM model achieves precision, recall and balanced accuracy scores of 0.84, 0.84, and 0.83, outperforming the probabilistic variant in precision and recall by 0.06 and 0.06, respectively. This experiment illustrates how PeakTK can be used to quickly compare the performance of different approaches on the same prediction task.

Next, we analyze and compare the performance of three peak day of the year algorithms that are made available in PeakTK. Figure 6b depicts the performance of these three algorithms i.e. CPP, stopping, and probabilistic approaches. For this experiment, we use the ISO-NE dataset, and perform prediction for the year 2020. For all three algorithms, we set $n = 10$ — i.e., our goal is to predict the 10 days of the year that experience the highest peak demand. The figure shows that the probabilistic approach achieves precision, recall, and balanced accuracy scores of 0.6, 0.6, and 0.79, respectively. The figure also shows that the stopping approach achieves precision, recall, and balanced accuracy scores of 0.78, 0.7, and 0.84, and outperforms the probabilistic approach by 0.18, 0.1, and 0.05 respectively. Finally, the figure shows that the CPP approach outperforms the other two algorithms for this task, achieving precision, recall, and balanced accuracy scores of 0.8, 0.8, and 0.89 respectively.

We then compare the performance of three peak day of the month prediction algorithms implemented in PeakTK — i.e., extreme temperature, VPeak, and the probabilistic approach. For this experiment, we use the ISO-NE dataset, and set $n = 3$ and $\Delta = 3$ — i.e., our goal is to predict the three days in a month that experience the highest peak demand by first finding the 6 peak days of the month and selecting the three highest days as a subset. Figure 7a depicts the results of this analysis. The figure shows that VPeak outperforms the other two algorithms, achieving precision, recall, and balanced accuracy scores of 0.45, 0.61, and 0.76, respectively. This is consistent with the observations made in the original work where VPeak was compared with the other two approaches in similar tasks. The figure also indicates that the probabilistic approach achieves higher recall and balanced accuracy scores (0.61 and 0.74 vs 0.48 and 0.68 respectively) compared to the extreme temperature approach. However, the precision score is significantly lower than the extreme temperature approach. This indicates that the two algorithms may be suitable for different prediction scenarios depending on the importance of each metric, and demonstrates how PeakTK can be used to aid in the making of such decisions.

To further analyze the performance of VPeak on peak days of the month prediction, we vary the number of predicted days and examine the resulting effect on precision, recall, and accuracy scores.

[8] https://darksky.net/dev
[9] https://www.tomorrow.io/weather-api/

	ISO-NE	ESO	Smart*
Type of data	Grid	Grid	Households
Data granularity	Hourly	30-minute	15-minute
Weather data granularity	Hourly	Hourly	Hourly
Duration	2018 to 2020	2016 to 2020	2015 to 2016
Demand forecast	3-day ahead	14-day ahead	None
Location	New England, US	Great Britain	MA, USA

Table 3: Summary of PeakTK's reference datasets

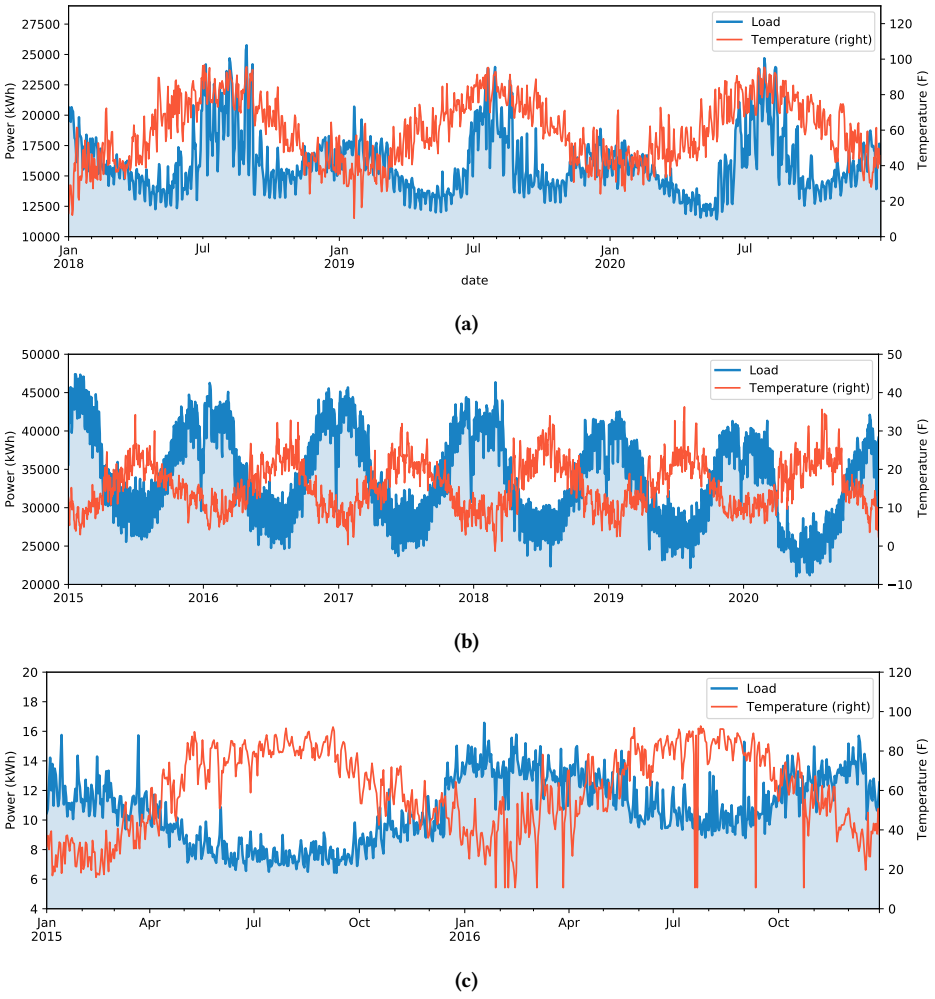


Figure 3: (3a) ISO New England demand and associated temperature from January 2018 to December 2020, (3b) NationalGrid ESO demand and related associated temperature from January 2015 to December 2020, and (3c) Smart* households demand and associated temperature from January 2015 to December 2016

For this experiment, we use the same ISO-NE dataset and vary n from 1 to 3. We set $\Delta = 3$ — i.e., we predict $n + 3$ days for each value of n. We then analyze the precision, recall, and balanced accuracy scores for each value of n. Figure 7b depicts the results of this analysis. The figure shows that as the value of n increases, precision, recall and balanced accuracy scores also increase. This is because as the value of n increases, the probability of selecting a day that is part of the peak days of the month also increases, and the algorithm performs better for higher values of n. This observation is consistent with the experiments performed in the original work, and shows how PeakTK can be used to enable the reproducibility of experimental results for various algorithms.

```
import peaktk

# load ISO-NE data
ISONE_data = peaktk.dataloader.ISONE_helper.load_data(DEMAND_PATH, WEATHER_PATH)

# data preprocessing and preparation
ISONE_data_scale, y_scaler = peaktk.preprocessing.scale(ISONE_data, min=0, max=1)
X, y, idx = peaktk.dataloader.ISONE_helper.create_dataset(ISONE_data_scale)
X_train, X_test, y_train, y_test, idx_train, idx_test
          = peaktk.preprocessing.train_test_split(X, y, idx, test_size=366)

# train demand prediction
lstm_model = peaktk.demand_prediction.lstm_dp.LSTMDemandPrediction()
lstm_model.fit(X_train, y_train, epochs=1000, verbose=1)

# predict peak demand
y_hat = lstm_model.predict(X_test)
pred_demand = y_scaler.inverse_transform(y_hat).flatten()

# set the number of peak days to predict and delta day
num_peakday=3, delta=3

# create and train peak day prediction model from historical load
vpeak_model = peaktk.peakday_prediction_monthly.vpeak.VPeak_Selection()
vpeak_model.train(y_train, idx_train, top_k=num_peakday+delta)

# predict peak day one day at a time
y_pred_peak = []
for i in range(len(pred_demand)):
    curr_date = idx_test[i]
    curr_pred_demand = pred_demand=[i]
    is_peak = vpeak_model.predict_ispeak(curr_date, curr_pred_demand)
    y_pred_peak.append(is_peak)

# find groundtruth peak day
y_test_peak = peaktk.stats.peakday_helper.identify_monthly_peak(ISONE_data, top_k=num_peakday)

# evaluate the results
precision = peaktk.metrics.metrics.precision(y_test_peak, y_pred_peak)
recall = peaktk.metrics.metrics.recall(y_test_peak, y_pred_peak)
balanced_accuracy = peaktk.metrics.metrics.balanced_accuracy(y_test_peak, y_pred_peak)
```

Figure 4: Example usage of the PeakTK's pipeline

Next, we analyze and compare the performance of various algorithms implemented in PeakTK across different time scales. To do so, we perform peak day of the month prediction on the ISO-NE dataset from the year 2020 using three algorithms for all months of the year. We set $n = 3$ and $\Delta = 3$ — i.e., our goal is to predict the top 3 days with the highest peak with 3 extra comparative days. For each algorithm, we analyze the month-on-month prediction performance, as well as compare the overall performance of each algorithm with each other. Figure 8a depicts the results of this analysis. The figure shows the applicability of different algorithms during different months of the year. For instance, the extreme temperature approach missed all peak days in January, April, and June. This is because the algorithm is based on temperature data only, and months that fail to experience extreme temperature relative to other days of the month may fail to trigger the algorithm's threshold. The figure also shows that since VPeak takes into account both temperature and load data, its performance is consistent throughout the year. The average recall for VPeak across all months of the year is 0.61. This experiment demonstrates the ability of PeakTK to identify applicable algorithms across varying time scales.

Lastly, we compare the performance of peak day of the year prediction algorithms implemented in PeakTK. To do so, we perform peak day prediction for 4 different years i.e., 2017-2019 using the ESO dataset. We use the first two years of the ESO dataset i.e. 2015 and 2016 for training, and the rest for our prediction task. For each year, we use the three peak day of the year prediction algorithms implemented in PeakTK and compare their recall across the whole period. We set $n = 10$ — i.e., our goal is to predict the top 10 days in each year that experience the highest load demand. Figure 8b depicts the results of this analysis. The figure shows that the recall for each algorithm varies from year to year e.g., CPP's recall is lower in the first two years than in the last two. The figure also shows the difference in year-to-year consistency across algorithms e.g., the probabilistic approach is more consistent in recall year-on-year compared to the other two algorithms. This analysis shows how PeakTK can be used to compare the overall performance of various algorithms across different time scales. In this case, the probabilistic

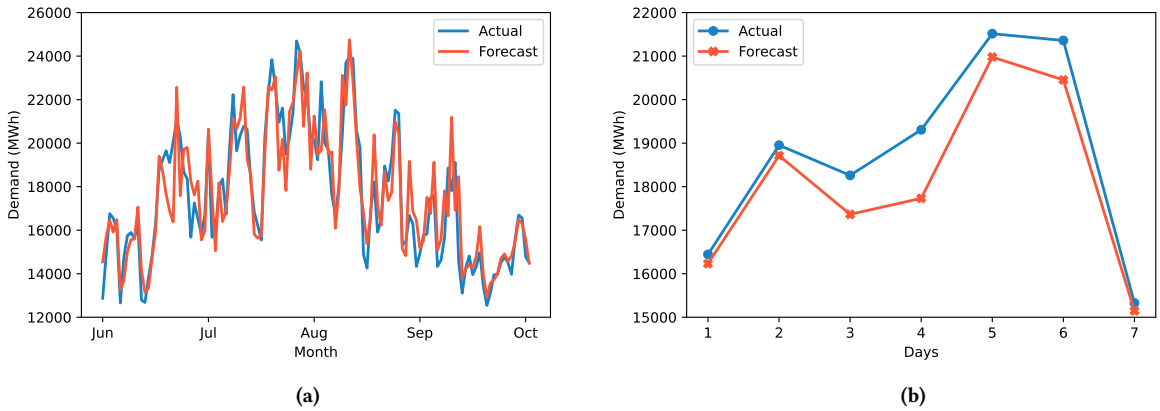


Figure 5: (5a) Actual and predicted demand using one of the demand prediction algorithms in PeakTK, and (5b), Comparison of actual and predicted demand over one week period

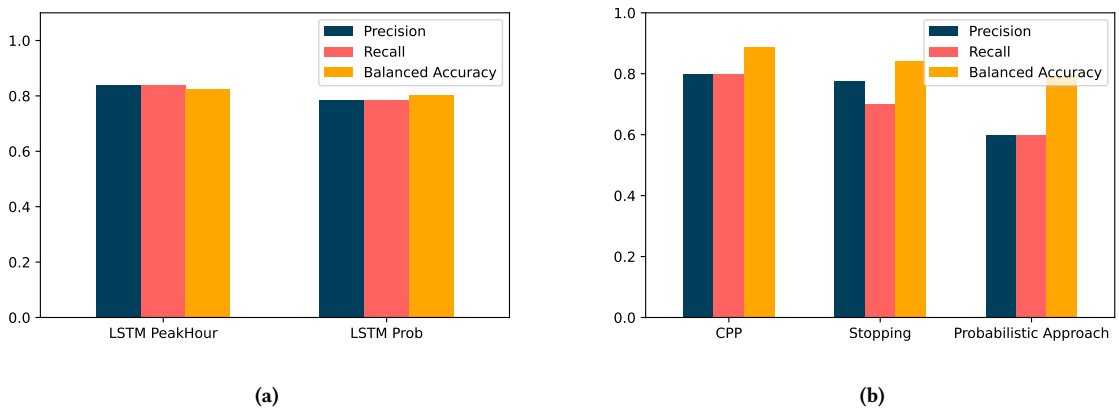


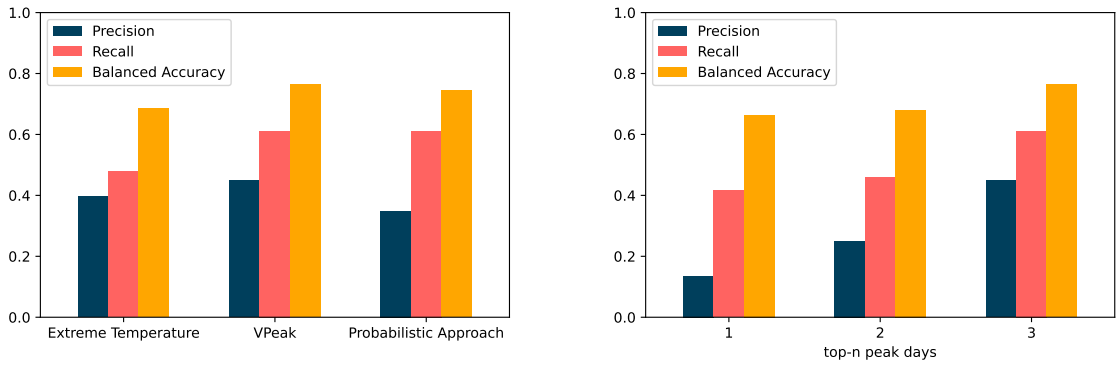
Figure 6: (6a), Comparison of precision, recall and accuracy of the two peak hours of the day prediction algorithms in PeakTK, and (6b) Performance of peak days of the year prediction algorithms implemented in PeakTK.

approach outperforms the other two approaches with an average recall of 0.45 across the 4 year period.

4.3 Battery scheduling

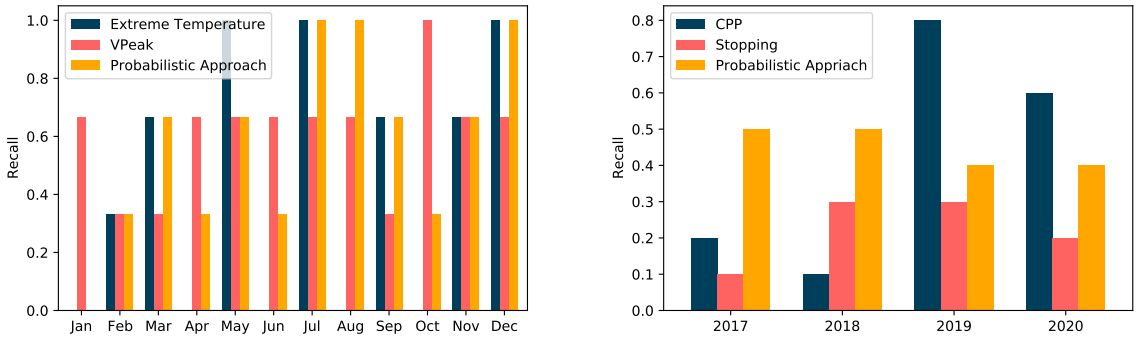
Peak shaving is a form of grid energy optimization that reduces the load on the grid during durations of peak demand. Utility companies deploy resources such as energy storage batteries in the distribution grid and operate them during peak demand hours. To perform peak shaving efficiently, utility companies must first estimate *when* the peak demand will occur, and *how much*, i.e., magnitude, of demand during the predicted peak period. This enables the utility to size the required resources, e.g., battery size, that must be dispatched during the peak demand hours in order to curtail the peak load. Since energy storage provides a finite amount of energy that can be depleted, it is important that utility companies accurately predict the peak demand so they can dispatch stored energy only during peak hours and recharge the grid batteries during off-peak hours.

We now show how PeakTK can be used to determine when energy storage should be dispatched for peak shaving. To do so, we use the ISO New England dataset included in PeakTK. We assume that the utility has installed 2000MWh energy storage in the grid to be used for peak load reduction. We then use PeakTK's peak demand prediction module to forecast next day energy demand for a period of 1 year. For each peak hour of the day, we dispatch the limited energy in the battery and recharge the battery during off-peak hours. Figure 9a depicts the true demand and predicted energy demand for a sample day from the dataset using the LSTM-based hourly demand prediction technique implemented in PeakTK. The figure shows that 4-8 PM is the peak hour of the day, and this is the hour during which energy storage should be dispatched. We perform this prediction for all days of the year while discharging the battery during peak hours and charging during off-peak hours. Figure 9b shows the original demand and the demand after dispatching the battery for peak shaving for one week. For the whole



(a) (b)

Figure 7: (7a), Performance comparison of the peak days of the month prediction algorithms implemented in PeakTK with n=3 and Δ=3, and (7b) Comparison of predicting different top-n peak days of the month using VPeak with Δ=3.



(a) (b)

Figure 8: (8a), Performance of peak day of the month algorithms over month using ISO-NE dataset, and (8b) Performance of peak day of the year algorithms over year using ESO dataset.

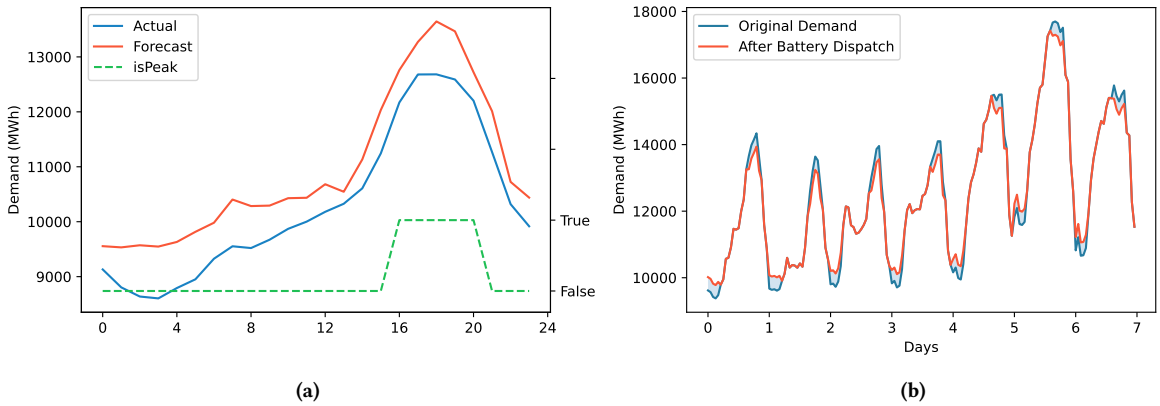
duration, we show that peak load can be shaved by using PeakTK to judiciously deploy energy storage during peak hours and recharge during off-peak hours. As can be seen in the figure, 2.2-2.8% of the peak can be shaved using this technique which reduces daily peak demand up to 400MWh.

5 CONCLUSIONS

Since the design of peak forecasting methods that predict the magnitude and time window of peak demand are at the heart of many energy optimization approaches, in this paper, we presented PeakTK, an open tool and reference datasets peak forecasting in energy systems. PeakTK implements a range of peak forecasting methods that have been proposed recently and exposes them through interfaces and library modules. We discussed the design and implementation of PeakTK and the various forecasting methods that are implemented by the tool. In the current release, PeakTK implements probabilistic, time-series-based and machine learning-based forecasting methods. Our framework is extensible and allows new algorithms and reference datasets to be added to it by the broader community. Finally, we presented case studies to demonstrate how PeakTK can be used for forecasting or quantitative comparisons of energy optimization methods. Our case studies illustrated how PeakTK can be employed for quantitative comparisons of forecasting methods and be used in higher-level energy optimizations such as battery scheduling. As future work, we plan to add support for other open datasets into PeakTK and provide sample code to illustrate typical use cases.

ACKNOWLEDGMENTS

We thank our anonymous reviewers for their insightful comments. This research is supported by NSF grants 2136199, 2021693, 2020888, and 2105494.



(a) (b)

Figure 9: (9a), Peak hours of the day prediction for a sample day, and (9b) Peak shaving performed by dispatching energy storage during predicted peak hours.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. https://www.tensorflow.org/ Software available from tensorflow.org.

[2] Abdullah Al Zishan, Moosa Moghimi Haji, and Omid Ardakanian. 2020. Adaptive control of plug-in electric vehicle charging with reinforcement learning. In *Proceedings of the Eleventh ACM International Conference on Future Energy Systems*. 116–120.

[3] Moshe Babaioff, Nicole Immorlica, and Robert Kleinberg. 2007. Matroids, secretary problems, and online mechanisms. (2007).

[4] Sean Barker, Aditya Mishra, David Irwin, Emmanuel Cecchet, Prashant Shenoy, Jeannie Albrecht, et al. 2012. Smart*: An open data set and tools for enabling research in sustainable homes. *SustKDD, August* 111, 112 (2012), 108.

[5] Sean Barker, Aditya Mishra, David Irwin, Prashant Shenoy, and Jeannie Albrecht. 2012. Smartcap: Flattening peak electricity demand in smart homes. In *2012 IEEE International Conference on Pervasive Computing and Communications*. IEEE, 67–75.

[6] Noman Bashir, Dong Chen, David Irwin, and Prashant Shenoy. 2019. Solar-TK: A Data-driven Toolkit for Solar PV Performance Modeling and Forecasting. In *2019 IEEE 16th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. IEEE, 456–466.

[7] Nipun Batra, Jack Kelly, Oliver Parson, Haimonti Dutta, William Knottenbelt, Alex Rogers, Amarjeet Singh, and Mani Srivastava. 2014. NILMTK: An open source toolkit for non-intrusive load monitoring. In *Proceedings of the 5th international conference on Future energy systems*. 265–276.

[8] Phuthipong Bovornkeeratiroj, John Wamburu, David Irwin, and Prashant Shenoy. 2021. VPeak: Exploiting Volunteer Energy Resources for Flexible Peak Shaving. In *International Conference on Systems for Energy-Efficient Buildings, Cities, and Transportation*.

[9] Yao Cheng, Chang Xu, Daisuke Mashima, Vrizlynn LL Thing, and Yongdong Wu. 2017. PowerLSTM: power demand forecasting using long short-term memory neural network. In *International Conference on Advanced Data Mining and Applications*. Springer, 727–740.

[10] Naveen Kumar Chikkakrishna, Chitirala Hardik, Kancherla Deepika, and Narendula Sparsha. 2019. Short-term traffic prediction using sarima and FbPROPHET. In *2019 IEEE 16th India Council International Conference (INDICON)*. IEEE, 1–4.

[11] Delson Chikobvu and Caston Sigauke. 2012. Regression-SARIMA modelling of daily peak electricity demand in South Africa. *Journal of Energy in Southern Africa* 23, 3 (2012), 23–30.

[12] Sajjad Fattaheian-Dehkordi, Alireza Fereidunian, Hamid Gholami-Dehkordi, and Hamid Lesani. 2014. Hour-ahead demand forecasting in smart grid using support vector regression (SVR). *International transactions on electrical energy systems* 24, 12 (2014), 1650–1663.

[13] Davide Frazzetto, Bijay Neupane, Torben Bach Pedersen, and Thomas Dyhre Nielsen. 2018. Adaptive user-oriented direct load-control of residential flexible devices. In *Proceedings of the Ninth International Conference on Future Energy Systems*. 1–11.

[14] Sean H. Gallagher. [n. d.]. Electric Demand Response Tariffs Clean-Up. https://www.pge.com/nots/rates/tariffs/tm2/pdf/ELEC_3221-E.pdf. (Accessed on 10/26/2021).

[15] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature* 585, 7825 (Sept. 2020), 357–362. https://doi.org/10.1038/s41586-020-2649-2

[16] Kaijian He, Lei Ji, Chi Wai Don Wu, and Kwok Fai Geoffrey Tso. 2021. Using SARIMA–CNN–LSTM approach to forecast daily tourism demand. *Journal of Hospitality and Tourism Management* 49 (2021), 25–33.

[17] Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. 2018. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*. 387–395.

[18] Yuheng Helen Jiang, Ryan Levman, Lukasz Golab, and Jatin Nathwani. 2014. Predicting peak-demand days in the Ontario peak reduction program for large consumers. In *Proceedings of the 5th international conference on Future energy systems*. 221–222.

[19] Anish Jindal, Mukesh Singh, and Neeraj Kumar. 2018. Consumption-aware data analytical demand response scheme for peak load reduction in smart grid. *IEEE Transactions on Industrial Electronics* 65, 11 (2018), 8993–9004.

[20] Guillaume Lemaître, Fernando Nogueira, and Christos K Aridas. 2017. Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research* 18, 1 (2017), 559–563.

[21] LinkedIn. [n. d.]. luminol - a light weight python library for time series data analysis. https://github.com/linkedin/luminol. (Accessed on 02/27/2022).

[22] Jinxiang Liu and Laura E Brown. 2019. Prediction of Hour of Coincident Daily Peak Load. In *2019 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. IEEE, 1–5.

[23] Priyanka Mary Mammen, Hareesh Kumar, Krithi Ramamritham, and Haroon Rashid. 2018. Want to reduce energy consumption, whom should we call?. In *Proceedings of the Ninth International Conference on Future Energy Systems*. 12–20.

[24] Aditya Mishra, David Irwin, Prashant Shenoy, and Ting Zhu. 2013. Scaling distributed energy storage for grid peak reduction. In *Proceedings of the fourth international conference on Future energy systems*. 3–14.

[25] Earo Wang. Mitchell O'Hara-Wild, Rob Hyndman. [n. d.]. Forecasting Models for Tidy Time Series - fable. https://fable.tidyverts.org/. (Accessed on 02/27/2022).

[26] The pandas development team. 2020. *pandas-dev/pandas: Pandas*. https://doi.org/10.5281/zenodo.3509134

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 8024–8035. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[29] Azar Rahimi, M Zarghami, M Vaziri, and S Vadhva. 2013. A simple and effective approach for peak load shaving using Battery Storage Systems. In *2013 North American Power Symposium (NAPS)*. IEEE, 1–5.

[30] Skipper Seabold and Josef Perktold. 2010. statsmodels: Econometric and statistical modeling with python. In *9th Python in Science Conference*.

[31] Akhil Soman, Amee Trivedi, David Irwin, Beka Kosanovic, Benjamin McDaniel, and Prashant Shenoy. 2020. Peak Forecasting for Battery-based Energy Optimizations in Campus Microgrids. In *Proceedings of the Eleventh ACM International Conference on Future Energy Systems*. 237–241.

[32] Sean J Taylor and Benjamin Letham. 2018. Forecasting at scale. *The American Statistician* 72, 1 (2018), 37–45.

[33] Youlong Yang, Jinxing Che, Chengzhi Deng, and Li Li. 2019. Sequential grid approach based support vector regression for short-term electric load forecasting. *Applied energy* 238 (2019), 1010–1021.

[34] Shizhen Zhao, Xiaojun Lin, and Minghua Chen. 2015. Peak-minimizing online EV charging: Price-of-uncertainty and algorithm robustification. In *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2335–2343.

[35] Yue Zhao, Zain Nasrullah, and Zheng Li. 2019. PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research* 20, 96 (2019), 1–7. http://jmlr.org/papers/v20/19-011.html