March 2019

Scalable Data-driven Modeling and Analytics for Smart Buildings

Srinivasan Iyengar

SCALABLE DATA-DRIVEN MODELING AND ANALYTICS FOR SMART BUILDINGS

A Dissertation Presented

by

SRINIVASAN IYENGAR

Submitted to the Graduate School of the
University of Massachusetts Amherst in partial fulfillment
of the requirements for the degree of

DOCTOR OF PHILOSOPHY

February 2019

College of Information and Computer Sciences

SCALABLE DATA-DRIVEN MODELING AND ANALYTICS FOR SMART BUILDINGS

A Dissertation Presented

by

SRINIVASAN IYENGAR

Approved as to style and content by:

Prashant Shenoy, Chair

David Irwin, Member

Deepak Ganesan, Member

Daniel Sheldon, Member

James Allan, Chair of Department
College of Information and Computer Sciences

DEDICATION

To my late father.

ACKNOWLEDGMENTS

Pursuing a doctoral degree is like a marathon that tests your emotional, physiological, and physical endurance. I was in no way prepared for it when I joined the Ph.D. program at the UMass CS department in the Fall of 2013. Although arduous, it was an extremely gratifying experience that I will cherish for a lifetime. Several individuals have played a pivotal role in my successful completion of this journey.

First and foremost, I would like to acknowledge the support and guidance offered by my advisor Prof. Prashant Shenoy. Over the past 5+ years, I have tried to learn from his clear thinking in fleshing out the problems and critically looking at the possible solutions. Moreover, he has provided constant encouragement to persevere on the topics covered in this thesis. At the same time, he has never shied away from any constructive criticism. However, his valuable feedback was necessary for my growth as a computer science researcher. He has always looked out for my long-term aspirations and helped me to achieve them. I have immense gratitude towards him and do not have enough words to express it adequately. Prof. David Irwin has been a great mentor and was always open to discussing my research. I have collaborated with David several times and am incredibly grateful for it. Prof. Deepak Ganesan has given valuable feedback on my research and helped me shape my thesis direction. Prof. Daniel Sheldon has always shown a keen interest in my research. Moreover, I collaborated with him on a significant project, which was part of my thesis. I am thankful for his guidance throughout my Ph.D.

Before starting my Ph.D. degree, I worked at Tata Research Development and Design Centre (TRDDC) in Pune for almost six years. I started as an undergrad intern working with Dr. Panduranga Rao and Dr. Sachin Lodha. The internship provided a peek at the life of a researcher, and I wisely decided to join TRDDC as a full-time employee. I would also

like to thank Prof. Harrick Vin for providing an opportunity to work at TRDDC as a full-time employee. Also, I was lucky to have Dr. Sachin Lodha as my supervisor. His calm demeanor and inspiring leadership were supremely helpful throughout my time at TRDDC. I also had the pleasure of working with Dr. Shirish Karande. His tremendous zeal made collaborating with him super fun. I also worked closely with Vijayanand Banahatti, Nikhil Patwardhan, Asim Roy, Dr. Panduranga Rao, Dr. John Augustine, Dr. Dilys Thomas, Dr. Sasanka Rao — all of whom provided tremendous guidance.

During the summer of 2017, I interned at Nokia Bell Labs located in Naperville, Illinois. The ten weeks I spent working with my supervisor Prof. Vijay Gurbani were quite outstanding. Prof. Gurbani continues to provide great mentorship and advice on a wide variety of topics such as my research, career after Ph.D., etc. At Bell Labs, I enjoyed working on a cool project and also forged a close friendship with a fellow intern - Heng Zhang (Ph.D. student at Purdue).

Doctoral study is a lonely endeavor. However, friends and colleagues make it a pleasurable experience. I was lucky to join the Ph.D. program alongside my dear friend Stephen Lee. Apart from providing a sink for my frustrations, he has been a great collaborator on almost all my projects. His energy on working on problems is unmatched — a quality I have tried to imbibe in me. There are countless instances where I have sought his counsel on various research projects. It is impossible to imagine my Ph.D. journey without his presence. Prateek Sharma has been an outlet for my never-ending soliloquies on diverse topics. Highlighted by his unique cynicism, the numerous discussions that I have had with him were outright hilarious. At the same time, he was always there to provide any help in my research. Specifically, I would like to thank him for his advice while preparing for my job search. Abhyuday Jagannatha has given great advice, and I would value his friendship for life. His unusual take on different matters has shaped my perspective. Collaborating with Navin Sharma on my first project at UMass was awesome. I am grateful for his mentorship. My sincere thanks to all my friends and colleagues at UMass for

providing great company: Venkatesh Kashyap, Amee Trivedi, Aditya Misra, Tian Guo, Xin He, Sandeep Kalra, Anushree Ghosh, Rishikesh Jha, Menghong Feng, John Wamburu, Phuthipong Bovornkeeratiroj, Assan Toleuov, Vani Gupta, Lucas Chaufournier, Bin Wang, and Sean Barker.

As per an old African proverb - “It takes a village to raise a child”. Similarly, I felt like the whole UMass community was instrumental in helping me with my doctoral studies. The staff at various UMass organizations such as the UHS, North Village Housing, CICS have made my Ph.D. easier. Special thanks Leeanne Leclerc and Karen Sacco for taking care of administrative tasks and paperwork.

Finally, I would like to thanks my family, i.e., my wife, mother, grandmother, and sister for their unwavering encouragement and support. During the rough periods in life, one needs to rechannel and strengthen their resolve. My family has always been there to remind me of this fact. My wife (Manasa Pidatala) has been on my side and has been my biggest champion. Her enthusiasm is contagious, and I would not have pursued my Ph.D. with the required focus without her. Her unparalleled support was not only necessary but also sufficient for carrying me through the trials and tribulations of doctoral studies.

ABSTRACT

SCALABLE DATA-DRIVEN MODELING AND ANALYTICS FOR SMART BUILDINGS

FEBRUARY 2019

SRINIVASAN IYENGAR

B.TECH., COLLEGE OF ENGINEERING PUNE

M.S., UNIVERSITY OF MASSACHUSETTS AMHERST

PH.D., UNIVERSITY OF MASSACHUSETTS AMHERST

Directed by: Professor Prashant Shenoy

Buildings account for over 40% of the energy and 75% of the electricity usage. Thus, by reducing our energy footprint in buildings, we can improve our overall energy sustainability. Further, the proliferation of networked sensors and IoT devices in recent years have enabled monitoring of buildings to provide data at various granularity. For example, smart plugs monitor appliance level usage inside the house, while solar meters monitor residential rooftop solar installations. Furthermore, smart meters record energy usage at a grid-scale.

In this thesis, I argue that data-driven modeling applied to the IoT data from a smart building, at varying granularity, in association with third party data can help to understand and reduce human energy consumption. I present four data-driven modeling approaches — that use sophisticated techniques from Machine Learning, Optimization, and Time Series Analysis — applied at different granularities.

First, I study IoT devices inside the house and discuss an approach called NIMD that automatically models individual electrical loads found in a household. The analytical model resulting from this approach can be used in several applications. For example, these models can improve the performance of NILM algorithms to disaggregate loads in a given household. Further, faulty or energy-inefficient appliances can be identified by observing deviations in model parameters over its lifetime.

Second, I examine data from solar meters and present a machine learning framework called SolarCast to forecast energy generation from residential rooftop installations. The predictions enable exploiting the benefits of locally-generated solar energy.

Third, I employ a sensorless approach utilizing a graphical model representation to report city-scale photovoltaic panel health and identify anomalies in solar energy production. Immediate identification of faults maximizes the solar investment by aiding in optimal operational performance.

Finally, I focus on grid-level smart meter data and use correlations between energy usage and external weather to derive probabilistic estimates of energy, which is leveraged to identify the least efficient buildings from a large population along with the underlying cause of energy inefficiency. The identified homes can be targeted for custom energy efficiency programs.

TABLE OF CONTENTS

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

CHAPTER 1

INTRODUCTION

Access to energy has enabled significant improvement in the quality of life in modern societies. Of the overall energy usage, almost 40% is consumed by buildings, ahead of industry and transportation sectors. Further, over the past several years, there has been an increased deployment of networked devices — also popularly known as the Internet of Things (IoT) devices — in built environments to monitor our energy generation and consumption. This thesis explores the challenges and opportunities in leveraging data produced from these IoT devices to infer actionable insights for better energy management by applying several techniques from Machine Learning, Optimization and Time Series Analysis.

1.1 Motivation

Energy, in different forms, has been vital for economic growth and human development. Modern energy services such as electricity, natural gas, cooking fuel have been shown to be directly linked to improved health and education. Over the past century, improved access to energy has been a significant factor in diminishing poverty levels. UN reports have identified a strong positive correlation between per capita energy consumption and Human Development Index [47].

However, increased energy consumption has unintended consequences. Majority of our energy sources are carbon-based fuels, causing a significant jump in CO_2 concentration in the atmosphere. This increase has lead to adverse impact on the environment and policymakers around the world are striving towards a sustainable energy future. Most of

the countries are focussing their efforts on reducing energy intensity from carbon-based sources [87]. Further, buildings account for over 40% of the energy and 75% of the electricity usage [61]. This energy consumption in buildings also accounts for 39% of the total CO_2 emissions. Hence, reducing our energy footprint in buildings has emerged as one of the most critical problems facing us today. However, there is another emerging trend that makes buildings an ideal candidate to focus our energy management measures, i.e., the proliferation of IoT devices.

IoT devices are being installed in built environments at an increasing rate. Advanced metering infrastructure in smart grids, also known as smart meters, can monitor a building's energy usage at fine-grained intervals. In the US alone, 70 million smart meters have been installed in 2016. This number is expected to increase to 90 million by 2020 [31]. Inside the house, smart plugs allow control over WiFi through APIs. NEST's programmable and self-learning WiFi-enabled thermostat has shown to reduce cooling and heating bills in residential buildings. Around 10% of the homes in the US ($\approx$ 12.7 million households) have smart devices deployed to varying degrees by 2015 [55]. Moreover, these IoT data can be combined with external data sources such as weather and real-estate data through APIs to drive deeper associations between our energy consumption and these factors.

We believe that this confluence of factors — i) increased monitoring and control infrastructure in built environments, and ii) availability of external data sources — permits use of sophisticated data analytics techniques to infer actionable insights. As more IoT devices in buildings become increasingly prevalent and ubiquitous, we argue that modeling energy usage patterns with external factors using data-driven modeling will be crucial in reducing our energy consumption. Figure 1.1 provides a basic pipeline for data-driven modeling. The data from IoT devices along with external 3rd party sources are ingested to perform some form of data transformation. The data transformation can be applied using techniques from different domains such as time-series techniques, optimization, and ma-

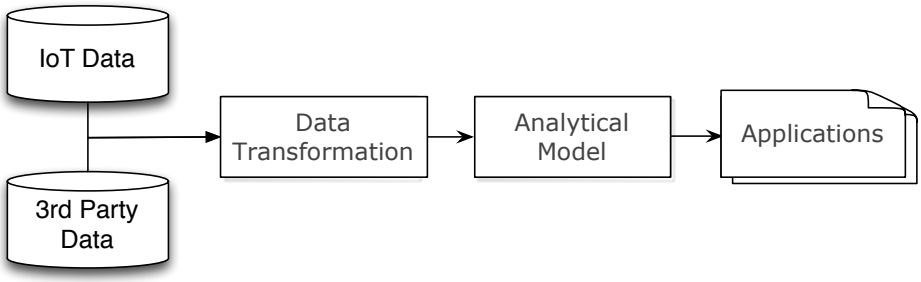


Figure 1.1. Data-driven modeling pipeline

chine learning. The data transformation produces an analytical model that can be used to build several applications that enable better energy management.

1.2 Thesis Contribution

As discussed earlier, increased deployments of networked sensors monitor the complete lifecycle of energy usage. Thus, data can be looked at different granularities that stretches from smart meters monitoring energy usage of homes in a city and solar meters monitoring residential rooftop solar installations to smart plugs monitoring appliance level usage inside the house. The thesis statement of my work is as follows - *data-driven modeling applied to the IoT data from a smart building, at varying granularity, in association with third party data can provide actionable insight to understand and reduce human energy consumption*.

By observing energy data at different granularities, I looked at addressing the following research questions using the data-driven modeling pipeline shown in Figure 1.1 -

- How are appliances within a house/building used? Can we model their energy consumption patterns?
- Can predict solar power generation from rooftop installations, using the minimal site-specific information to integrate renewable sources of energy in our daily lives?

Granularity	IoT Device data	3rd Party data	Data Transformation	Analytical Model	Applications
Multiple sensors inside the House	Plug Meters	*NA*	NIMD	Device profile and usage model	i) NILM ii) Faulty device identification
Grid-level	Electric & Gas Meters	i) Past Weather ii) Tax accessor	WattHome	Weather sensitivity	i) Directed energy audits ii) Targeted energy efficiency programs
Individual House	Solar generation Meters	i) Past generation ii) Past weather iii) Site Lat/Long	SolarCast	Solar power prediction	i) Flexible load scheduling ii) EV charging station
Individual House	Panel electrical properties, Panel physical properties, IR and Visible Camera etc.	i) Current weather	SolarSpy [Proposed Work]	Real-time Anomaly Detection	i) Early Maintenance ii) Improved Performance

Table 1.1. Summary of the work proposed in this thesis

- Can we identify anomalies in solar power generation to maximize renewable energy potential?
- Can we identify causes of inefficiency in residential buildings at city-scale to target energy efficiency programs?

The solution to each of these questions are the individual contributions presented in this thesis. Table 1.1 summarises these based on the data-driven modeling pipeline at varying granularity of the energy data. Below, we discuss each of them briefly.

1.2.1 Modeling consumption and usage patterns of electrical appliances

A variety of energy management and analytics techniques rely on models of the power usage of a device over time. Unfortunately, the models employed by these techniques are often simplistic, such as modeling devices as being on with a fixed power usage or off and consuming little power. The power usage of even relatively simple devices exhibits much more complexity than a simple on and off state. Moreover, the process involved manual intervention. To address the problem, in this thesis, we present a Non-Intrusive Model Derivation (NIMD) algorithm to automate modeling of residential electric loads. NIMD

automatically derives a compact representation of the time-varying power usage of any residential electrical load, including both the device's energy usage and its pattern of usage over time. Such models are useful in several applications — such as Non-Intrusive Load Monitoring, which has relied on simple on-off models in the past. Further, devices at the end-of-life can be identified by observing deviations from their historical device models.

1.2.2 Predicting solar power generation from rooftop installations

One can also apply data-driven modeling on the data from residential rooftop solar meters. Accurately forecasting solar generation is critical to fully exploiting the benefits of locally-generated solar energy by scheduling flexible loads. In this thesis, I present two machine learning techniques to predict solar power from publicly-available weather forecasts. We use these techniques to develop SolarCast, which automatically generates models that provide customized site-specific predictions of solar generation. SolarCast utilizes a "black box" approach that requires only i) a site's geographic location and ii) a *minimal* amount of historical generation data. Since we intend SolarCast for small rooftop deployments, it does not require detailed site- and panel-specific information, which owners may not know, but instead automatically learns these parameters for each site.

1.2.3 Detecting anomalies in solar power generation

Solar panel generation is subject to several factors. First, weather-related factors such as cloud cover cause intermittency in energy production, whereas higher temperatures reduce panel efficiency. Apart from these snow, soiling, and pollen block solar irradiance from falling on the panels thereby reducing generation. Moreover, power generation can also be reduced due to regular wear and tear causing cracks and/or discoloration of the panels. At the same time, one needs to distinguish between reduction in power output due to anomalies and other factors such as cloudy conditions. Thus, identifying anomalies by observing just the power produced is non-trivial. In this thesis, we propose SolarClique, a data-driven approach that can flag anomalies in solar power generation with high accuracy. Unlike prior

approaches, our work neither depends on expensive instrumentation nor does it require external inputs such as weather data. Rather this approach exploits correlations in solar power generation from geographically nearby sites to predict the expected output of a site and flag anomalies. Detecting these anomalous is crucial in maximizing the investment made to have a solar installation.

1.2.4 Identifying the inefficient homes along with its probable source

At a higher granularity, data-driven modeling can be applied at the grid-level by observing the smart meters recording electric and gas usage. In this thesis, I present an approach called WattHome that first disaggregate a building's observed energy usage into its heating, cooling and base components. Unlike past methods such as NILM, the model presented in this thesis does not require a priori training using ground truth data and instead uses correlations between energy usage and external weather to derive probabilistic estimates of energy use under different conditions. Next, this probabilistic model is used to identify the least efficient buildings from a large population and develop algorithms to diagnose the underlying cause of energy inefficiency. As shown in Table 1.1, WattHome produces a weather sensitive analytical model of individual households in a city that can be leveraged to conduct directed energy audits and correctly target energy efficiency programs.

1.3 Thesis Outline

We structure the remainder of this thesis as follows. **Chapter 2** provides background on data-driven modeling approaches applied at varying granularity and discusses prior work along with some of the existing challenges. **Chapter 3** describes Non-Intrusive Model Derivation (NIMD) algorithm that automates the modeling of electric loads. **Chapter 4** presents SolarCast, a black-box approach to automatically provide site-specific solar predictions. **Chapter 5** discusses SolarClique, a sensorless method to detect anomalies in solar power generation. **Chapter 6** explains WattHome, an approach to identify inefficient

homes along with the probable cause. Finally, **Chapter 7** presents the conclusions of this thesis.

CHAPTER 2

BACKGROUND AND RELATED WORK

This chapter provides background and related work on data-driven modeling used by observing human energy usage patterns. Specifically, we discuss a few that are applied at various granularities, i.e., from individual appliances in a house to its overall energy usage at a grid-level from individual smart meters recording gas and electric consumption. Further, we also extend our discussion to modeling renewable energy generation (primarily solar). We also present challenges not addressed in state of the art.

2.1 Data-driven Modeling at the Appliance-level

Data-driven Modeling at the finest granularity, i.e., at the appliance-level of our energy usage, allows detailing electricity consumption patterns and the running conditions of the individual electrical loads to the consumers. Below, we describe a fundamental appliance-level modeling approach, called Non-Intrusive Load Monitoring (NILM), that enables a wide variety of applications.

2.1.1 Non-Intrusive Load Monitoring

Non-Intrusive Load Monitoring (NILM) techniques decompose the total energy usage of a building into individual components — such as usage from lighting, from individual appliances such as TV or washing machine and AC, furnace and water heaters. NILM was introduced by Hart et. al. [50] to disaggregate building energy usage. NILM techniques have been well-studied for decades [21, 25, 67]. The key premise behind NILM is that each load or appliance exhibits unique power behavior ("power fingerprint") and that it is

possible to discern these patterns in the total energy usage and "extract" the power usage of individual loads. These techniques use pattern recognition, signal processing or machine learning techniques to perform load disaggregation.

Such techniques are becoming more commonplace with the growing popularity of Internet-of-Things (IoT) devices being deployed in smart homes. Modeling of electrical loads is a fundamental building block that is essential for driving higher-level techniques. Such models are compact representations of the electrical usage patterns exhibited by a load (e.g., a washing machine or TV) as well as temporal characteristics that describe when the load is used by residents (e.g., residents watch TV every evening and do laundry on weekends). NILM-based analytics approaches have been used in a variety of applications, such as inferring occupancy patterns [26,64], reducing peak demand by opportunistic load scheduling [20], and learning thermostat schedules [56].

2.1.2 Challenges

Many NILM approaches model electrical loads as simple *on-off* devices, where the load draws a fixed amount of power when turned on. As illustrated in Figure 2.1, which depicts a washing machine, most residential loads exhibit complex and varied power patterns that are distinct from the simple *on-off* behavior. Hence, disaggregating loads based on a simplistic and inaccurate understanding of a load's behavior significantly degrades the accuracy of higher-level techniques. Studies have observed that simplistic or coarse-grain models can be detrimental to the accuracy and effectiveness of higher-level approaches [17].

Despite its importance, empirical or analytic modeling of electrical loads has received relatively little attention. Typically, common devices, such as TVs, refrigerators, and computers, are only rated (often conservatively) based on their maximum power but include no details of how the device consumes power over time. A recent effort [18] analyzed empirical data gathered from a large number of residential loads to argue that only the simplest loads, such as light bulbs, exhibit a simple on-off behavior and demonstrates that most

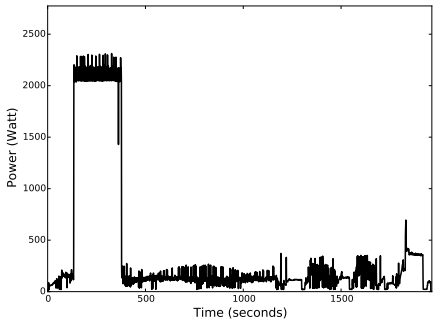


Figure 2.1. Observed power usage of a washing machine

loads exhibit more complex exponential decays or growth, bounded min-max, and cyclic patterns. While this work proposed more complex analytic models to describe load behavior, it did not propose any algorithms or approaches to derive (or construct) such models automatically. That is, it required manual modeling of a load by an expert before such a model could be used by higher-level optimizations. However, manual modeling of highly complex loads is time-consuming, and, for a load as complex as Figure 2.1, potentially infeasible given the load's complexity.

2.2 Data-driven Modeling in Renewable Energy Management

In this section, we primarily focus on solar energy — the predominant form of renewable energy deployed in smart buildings. Below, we provide a background on how solar installations in smart buildings (also called rooftop solar) differ from large utility-scale solar farms and the unique challenges they possess.

2.2.1 Rooftop vs Utility-scale Installations

Solar deployments come in a wide variety of sizes, ranging from the massive solar farms deployed by utilities (and some datacenter operators [11]) to small and medium-sized deployments deployed by homeowners, farmers, and local businesses. Overall, nearly half of aggregate solar capacity is now derived from small-scale home deployments ($<$10kW), many of which rely on *net metering* to transfer surplus energy to the grid [74] — thereby eliminating the need for expensive battery-based energy storage. As the number of home

deployments grows, the need for predictive tools that provide near-term forecasts of solar generation at the time-scales of tens of minutes to days is becoming increasingly important. A detailed survey of solar power prediction techniques using custom models with known parameters can be found in Lorenz et al. [70] and Huang et al. [52]. Accurate near-term predictions, if available, have the potential to yield numerous benefits. For instance, homes that plan to better align their energy usage with solar generation can substantially decrease the surplus energy they contribute to the grid via net metering. Minimizing the energy contributed by net metering is important for two reasons.

- First, consuming power at the point of production is inherently more energy-efficient than net metering, since it eliminates transmission losses.
- Second, the increasing stochasticity in demand from net metered solar installations complicates utilities' task of balancing supply and demand in real time, since utilities cannot accurately account for home solar generation when planning generator dispatch schedules, i.e., when to activate and deactivate generators to ensure the supply of power matches the grid's net demand.

2.2.2 Challenges

Predicting solar generation for small-to-medium-sized solar deployments raises a different set of challenges than predicting it for massive solar farms. Specifically, the location of massive solar deployments is carefully chosen to be in open spaces that minimize occlusions. This enables installers to maximize solar output by precisely tuning the orientation of the panels or employing "trackers" that continuously change the tilt of the panels to track the sun. Further, industrial solar farm operators routinely clean the panels to keep them free from dust or snow to maintain optimal solar output. At the same time, industrial operators also have the technical expertise and resources to carefully design and tune custom models to predict the future solar output.

Unfortunately, the characteristics described above do not hold for most small-to-medium-scale solar deployments. For instance, the orientation and pitch of a home's roof constrain the installation of rooftop solar panels and limits the ability to optimize their placement. As a result, shadows from nearby objects, such as trees or even neighboring buildings, are common; these shadows complicate solar generation forecasting, as they change based on the time of the day and season of the year. Roofs are often not easily accessible, which also limits the ability to clean the panels. Finally, neither the owners nor the installers of small solar deployments typically have the technical expertise or the resources to develop custom prediction models that are specific to their setup. The large number of small-to-medium-scale deployments makes it challenging for technical experts to manually develop custom models for each site, as is common with industrial-scale solar farms. In fact, due to the factors above, since the models for small rooftop solar deployments are more complex and dynamic than for large solar farms, they require even more time and expertise to develop.

We note that most of the characteristics are highly specific to a particular installation. Thus, precisely quantifying them is not practical, or even possible, for owners of solar deployments having a limited technical background. Also, the aggregate amount of solar capacity installed at small residential solar deployments is expected to exceed that of utility-scale deployments for the first time in 2017 [1]. Thus, there is an increasing need for a black-box approach to predict the energy generated for rooftop solar installations that can automatically determine few of the unknown configurations parameters.

2.3 Data-driven Modeling in Solar Anomaly Detection

In this section, we focus on detecting anomalous solar power generation in a residential solar installation. Unlike power generation from traditional mechanical generators (e.g., diesel generators), where power output is constant and controllable, the instantaneous power output from a PV system is inherently intermittent and uncontrollable. The

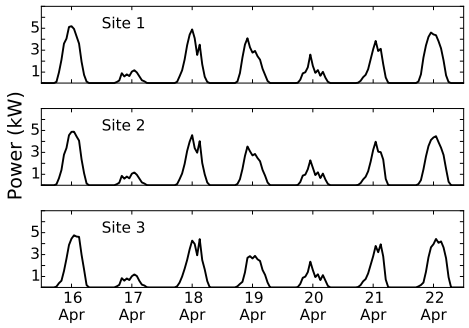


Figure 2.2. Power generation from three geographically nearby solar sites. As shown, the power output is intermittent and correlated for solar arrays within a geographical neighborhood.

solar power output may see sudden changes, with energy generation at peak capacity at one moment to reduced (or zero) output in the next period (see Figure 5.1). The change in the power output can be attributed to a number of factors and our goal is to determine whether the drop in power can be attributed to anomalous behavior in the solar installation.

2.3.1 Factors affecting solar output

A primary factor that influences the power generation of a solar panel is the solar irradiance, i.e., the amount of sunlight that is incident on the panel. The amount of sunlight a solar panel receives is dependent on many factors such as time of the day and year, dust, temperature, cloud cover, shade from nearby buildings or structures, tilt and orientation of the panel, etc. These factors determine the amount of power that is generated based on how much light is incident on the solar modules.

However, a number of other factors, related to hardware, can also reduce the power output of a solar panel. For instance, the power output may reduce due to defective solar modules, charge controllers, inverters, strings in PV, wired connections and so on. Clearly, there are many factors that can cause problems in power generation. Thus, factors affecting output can be broadly classified into two categories: (i) *transient* — factors that have a

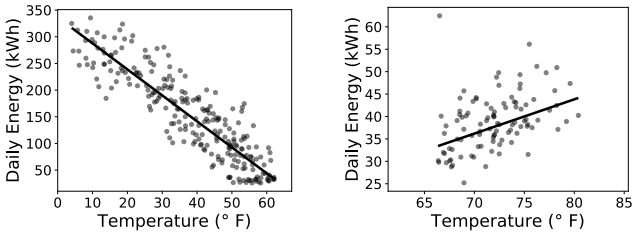
temporary effect on the power output (such as cloud cover); and (ii) *anomalies* — factors that have a more prolonged impact (e.g., solar module defect) on the power output.

The transient factors can further be classified into *common* and *local* factors. The common factors, such as weather, affect the power output of all the solar panels in a given region. Moreover, its effect is temporary as the output changes with a change in weather conditions. For instance, overcast weather conditions temporarily reduce the output of all panels in a given region. The local factors, such as shade from nearby foliage or buildings, are usually site-specific and do no affect the power output of other sites. These local factors may be recurring and reduce the power output at fixed periods in a day. In contrast, anomalous factors, such as bird droppings or system malfunctions, reduces power output for prolonged periods and usually require corrective action to restore normal operation of the site. Note that both transient and anomalous factors may reduce the power output of a solar array. Thus, a key challenge in designing a solar anomaly detection algorithm is to differentiate the reduction in power output due to transient factors and anomalies.

2.3.2 Challenges

Prior approaches have focused on using exogenous factors to predict the future power generation [15, 71, 94]. A simple approach is to use such prediction models and report anomaly in solar panels if the power generated is below the predicted value for an extended period. However, it is known that external factors such as cloud cover are inadequate to accurately predict power output from solar installations [58]. Thus, prediction models may over- or under-predict power generation, and such an approach may not be sufficient for detecting anomalies.

Prediction models can be improved using additional sensors but can be prohibitively expensive for residential setups [75]. For instance, drone-mounted cameras can detect occlusions in a panel but are expensive and require elaborate setup. Other studies use an ideal model of the solar arrays to detect faults [9, 35]. These studies rely on various site-



(a) Winter Months (b) Summer Months

Figure 2.3. Linear relationship between energy consumption and ambient temperature.

specific parameters and assume standard test condition (STC) values of panels are known. However, site-specific parameters are often not available. Thus, most large solar farms usually depend on professional operators to continuously monitor and maintain their setup to detect faults early[1][2]. Clearly, such elaborate setups may not be economically feasible in a residential solar installation. Thus, there is a need for a data-driven and cost-effective approach for detecting anomalies in a solar installation.

2.4 Data-driven Modeling at the Grid-level

Data-driven modeling at the grid-level using smart meters monitor energy usage of individual homes. This enables comparative analysis among the homes by using external sources of real-estate data describing building type, age, and square footage. Additionally, weather data can be utilized in association with energy consumption to identify inefficient homes. Below, we provide some background on energy inefficiency in buildings.

2.4.1 Energy inefficiency in buildings

Energy usage in residential buildings has different sources such as heating and cooling, lighting, household appliances and electronic equipment. There can be many causes of inefficiencies in each of these components, such as the use of inefficient incandescent light-

[1]ESA Renewables: `http://esarenewables.com/`

[2]Affinity Energy: `https://www.affinityenergy.com/`

ing and the use of inefficient (e.g., non-energy star) appliances. Studies have shown that heating and cooling is the dominant portion of a building's energy usage, comprising over half of the total usage [2, 61], and it follows that the most significant cause of inefficiency lies in problems with heating and cooling. Two factors determine heating and cooling efficiency of a building: (1) the insulation of the building's external walls and roof ("building envelope") and their ability to minimize thermal leakage, and (2) the efficiency of the heating and cooling equipment. Recent technology improvements have seen advancements on both fronts. New buildings are constructed using modern methods and better construction materials that yield a building envelope that minimizes air leaks and thermal loss through better-insulated walls and roofs and high-efficiency windows and doors. Similarly, new high-efficiency heating and AC equipment are typically 20-30% more efficient than equipment installed in the late 1990s and early 2000s.

Unfortunately, older residential buildings and even ones built two decades ago do not incorporate such energy efficient features. Further, the building envelope can deteriorate over time due to age and weather and so can mechanical HVAC equipment. Consequently, an analysis of a building's heating and cooling energy use can point to the leading causes of a building's energy inefficiency.

An approach for modeling a building's heating and cooling usage is to model its dependence on weather [106]. For example, a building's heating and cooling usage can be modeled as a linear function of external temperature. To intuitively understand this, consider cooling energy usage during the summer. The higher the outside temperature on hot summer days, the higher the AC energy usage. Since the difference between outside and inside temperatures are larger, there is higher thermal gain — requiring a longer duration of cooling to maintain a set indoor temperature. Thus, there is a linear relationship between heating and cooling energy use and outside temperature. Such linear models are commonly used in the energy science research and capture the relationship between energy use and the outside temperature, and we assume them in our work. Figure 2.3(a) and (b) illustrates

the linear dependence of heating energy used in the winter and cooling energy used in the summer on external temperature for an example home.

NILM-based methods that assume the availability of ground truth data can disaggregate loads and extract heating and cooling usage from the total energy usage and then build a model to correlate heating and cooling usage to parameters such as temperature. However, such an approach is infeasible to find the least efficient buildings from a population of hundreds or thousands of buildings, as it implies we need full information of every load in all buildings in a city. Instead of performing full disaggregation of loads, our approach focuses on *partial coarse-grain decomposition*. In contrast to full load disaggregation, one can decompose the total energy usage into two components—*weather-dependent* and *weather-independent* components. As noted earlier, weather-independent loads consist of heating and cooling equipment in a home, since energy usage of these appliances is dependent on external weather parameters such as temperature. Weather independent loads consist of all loads such as cleaning and cooking appliances, TV, electric equipment, lighting, etc. that do not depend on temperature.

2.4.2 Challenges

By simply using the insight that heating and cooling energy use has a linear dependence on temperature, one can construct a model that uses the correlations between the observed variations in energy and temperatures to decomposes the total usage into these two components. Consequently, one can automatically build models of heating and cooling usage (as a function of temperature) as well as the non-HVAC usage and overcome the limitations of full load disaggregation methods. Simple versions of such linear models have been used in energy science research for many years [41, 51, 63]. However, these models do not capture the stochastic variations in heating and cooling as well as the weather-independent energy usage resulting from day to day variations in human activities inside a home. For example, energy usage on the weekend may be higher since the building is occupied for longer

periods, increasing the heating and cooling usage as well as usage from activities such as washing clothes. Figure 2.3(a) and (b) depict the substantial variance in energy usage.

CHAPTER 3

AUTOMATED MODELING OF RESIDENTIAL LOADS

This chapter presents Non-Intrusive Model Derivation (NIMD), an algorithm to automate modeling of residential electric loads. We initially start with describing the analytical models that represent different characteristic loads found in a residential building followed by describing the NIMD algorithm in detail. We conclude with a detailed evaluation illustrating the efficacy of our algorithm.

3.1 Motivation

Modeling of electrical loads is an essential part of several higher-level techniques such as NILM. But several NILM approaches model electrical loads as simple *on-off* appliances, i.e., the devices consume a fixed amount of power when turned on. These models are quite simplistic and do not correctly explain the behavior of energy drawn by these devices. A recent effort [18] empirically characterized electrical loads from their power traces and demonstrated that common loads could be modeled analytically. However, it involved manual modeling of electrical loads by an expert and assumed that *a priori information* such as the type of device being known. Thus, there is a need to automatically derive the "best" analytic description that explains the observed behavior. Further, past work has not adequately looked at the way humans interact with the devices. Thus, apart from the modeling the *power signature*, the usage pattern of the devices can also be captured.

3.2 Analytical Characterization of Electrical Loads

Below, we show the empirically observed behavior of each basic load type modeled using one of four analytic equations:

1. *On-Off Model:* In this case, the load draws a fixed power P_{on} when active and zero or a small amount of standby power P_{off} when inactive. Simple resistive loads were found to exhibit such binary on-off behavior.

2. *On-Off Decay Model:* In this case, the power usage of the load exhibits an exponential decay behavior, represented as follows.

$$\mathbf{X}(t) = \begin{cases} p_{active} + (p_{peak} - p_{active})e^{-\lambda t}, & 0 \leq t < t_{active} \\ X_{off}, & t \geq t_{active} \end{cases} \tag{3.1}$$

   Here, p_{peak} represents the initial surge power, p_{active} is the stable power level and λ captures the rate of decay. Many inductive loads consisting of AC motors were shown to exhibit this behavior.

3. *On-Off Growth Model:* Some loads exhibit a growth behavior i.e. a logarithmic growth in power usage. We model such devices using a logarithmic function (inverse of the exponential function) that starts with a power level p_{base} with a growth parameter λ. We refer to such loads as an *on-off growth* model:

$$\mathbf{X}(t) = \begin{cases} p_{base} + \lambda \cdot \ln t, & 0 < t < t_{active} \\ X_{off}, & t \geq t_{active} \end{cases} \tag{3.2}$$

4. *Stable Min-Max* and *Random Range* Models: All non-linear loads exhibit a degree of random behavior and the observed behavior was characterized as a random walk between an upper and lower bound (referred to as random range) or a stable power draw with random upward or downward deviation (referred to as a stable min-max

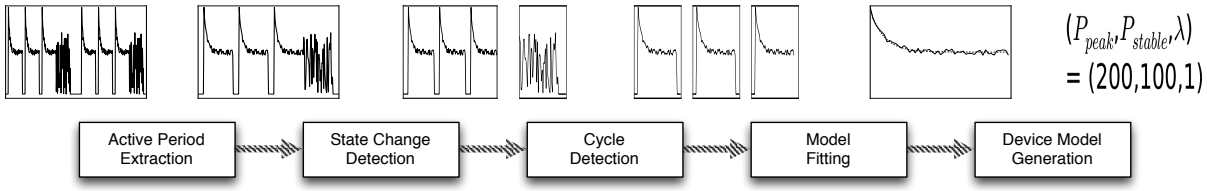


Figure 3.1. Pipeline of steps involved in device modeling of an electrical load

model). While prior work [18] employed a uniform distribution to model random power fluctuations in non-linear loads, our work uses the more general $Gamma$ and $LogGamma$ distributions to model stable min and stable max behavior with random deviations. The two models are shown below.

$$\mathbf{X}(t) \sim \ Gamma(\alpha, loc, scale), \quad 0 < t < t_{active} \tag{3.3}$$

$$\mathbf{X}(t) \sim \ LogGamma(\alpha, loc, scale), \quad 0 < t < t_{active} \tag{3.4}$$

Here, α, loc and $scale$ denote the shape, location and scale parameters for the two distributions. Electronic loads with switched-mode power supplies, such as TVs, phone chargers, and computers were shown to exhibit this behavior.

5. *Cyclic Model:* Any load that exhibits repeating patterns was characterized as cyclic with a certain period. All other complex loads that included multiple types of basic loads were characterized as a linear combination of above loads.

3.3 NIMD Algorithm

In this section, we propose our Non-Intrusive Model Derivation (NIMD) approach for automated modeling of electrical loads. Broadly our approach has two parts: (i) device modeling, where we learn the power usage behavior of the load when it is active, and (ii) usage modeling, where we learn how the users use the load in a particular environment. Although, both components are necessary to model the overall load behavior, they are

independent and can be used on their own for specific use-cases. In what follows, we describe the details of the device and usage modeling.

3.3.1 Basic Approach

Figure 3.1 depicts the high-level approach for NIMD device modeling. Given a raw power trace of a load, NIMD's approach to constructing a **device model** involves the following steps:

- **Step 1: Active period extraction:** For a given trace, the first step is to partition the trace into active and inactive periods. An *active* period is one where the load is operating and drawing power, while an *inactive* period is one where the load is turned off or in standby mode (and not in active use). A long power trace will consist of alternating periods of active and inactive use, and hence, this step extracts active periods from the trace.

- **Step 2: State change detection via change point detection:** During each active period, a load may transition through different active states and exhibit a different type of power variations in each state as it transitions from one active state to another. In this step, our technique uses a change detection algorithm to determine these state transitions, which manifest as "significant" changes in power behavior. By further partitioning an active period at each state transition, we obtain a set of trace segments corresponding to different active states within each active period.

- **Step 3: Cycle detection:** Next our technique compares the power patterns across states to determine if the behavior is cyclic. If a repeating pattern of state transitions is found, then a more compact model can be constructed by analyzing a repeating cycle rather than all trace segments.

- **Step 4: Model fitting:** In this key step, our technique tries to fit the trace segment extracted from each state onto various analytic models described in Section 3. The

best fit is then chosen, which yields both the load type seen during that active state and the parameters of the model describing that observed behavior.

- **Step 5: Device Model Generation:** The previous step yields a sequence of analytic models, one for each active phase, as well as cyclic dependencies, if any, for each active period. We repeat this process for each active period present in the trace. The final step then is to catalog the sequence of analytic functions for the overall model as well as the parameters of the various analytic functions found by our technique.

While the previous steps construct a device model from a raw power trace, we now describe the high-level approach for deriving a **usage model** for the load.

Intuitively, the usage model involves determining *how frequently* a load is used and *when* it is used (e.g., mornings, evenings, weekends, summer, etc.) To derive the usage, consider the first step of the device modeling, namely active period extraction. In this step, the trace is partitioned into active and inactive periods. In doing so, we obtain, over the period of the trace, start times of each active period, and the lengths of each active ("on") and inactive ("off") periods. This data is used to construct a usage model as follows:

- **Step 1:** Our technique first finds the shortest duration (e.g., a day, week, month or year) over which the load exhibits "similar" behavior. In order to derive a compact model, this is the period over which the usage of the device repeats in a statistically meaningful manner and captures the seasonality of the usage.

- **Step 2:** Next, our technique constructs probability distributions for the start times and the active and inactive period lengths for the above duration. The joint probability distribution of these variables yields the usage model.

Together, device and usage models together describe a compact model for residential electrical loads. Below, we discuss the key steps in device and usage modeling in detail.

3.3.2 Device Modeling

Figure 3.1 depicts the key steps for automated device modeling, which are outlined in the previous section. We discuss each step in more detail below.

- **Step 1: Active Period Extraction -** Each load alternates between active and inactive periods. Inactive periods can be determined by sequentially scanning the trace for periods where the power usage is less than a low threshold ϵ for durations longer than a threshold interval τ. This threshold corresponds to standby mode, the load will either consume zero power or a small amount of standby power (also known as "vampire" power [38]). Once inactive periods are labeled in the trace, the remaining periods are, by definition, active periods.

- **Step 2: State Change Detection -** When a load is active, it may transition between different active states. Each state may represent transitions between different basic loads that are components of the overall load, or may represent different active states of a basic load. Each state manifests itself in terms of a different power usage pattern. For example, a washing machine cycle may involve wash, rinse, and spin cycles, where different components of the washer (i.e., basic loads) activate in turn. Similarly, during the spin cycle, the motor may transition through different speeds, each of which is a distinct state with a different power usage level. Since each active state has a distinct and observable power usage pattern, our technique uses a *change point detection algorithm* to determine when significant changes (i.e., transitions) occur in the observed power usage. Change point detection (also known as change detection) is a well-known technique that is used for anomaly detection [81, 92]. However, since traditional change detection techniques are not well suited to our problem, we devise a new change detection algorithm to detect state transition points within an active period.

Our energy-specific change point detection algorithm is based on the notion of *approximate entropy*. Intuitively, entropy is a measure of the unpredictability of information content. In the context of time series data, *Approximate Entropy* (ApEn) is a technique to quantify unpredictability of fluctuations in data [82]. Our algorithm operates over a sliding window of the power time series for an active period. For each position of the sliding window, it computes the approximate entropy H over a the window of length ϕ [1]. Next, we need to detect *large* changes in approximate entropy as the window slides over the time series. To do so, we employ the *Canny Edge Detection* algorithm [23], a technique from computer vision, to detect "edges" where there are sudden changes in the entropy values H. Further, we remove certain edges that are within a predefined range δ of each other. Doing so yields instants in the power trace where significant changes in approximate entropy (which represent active state changes) are observed. Algorithm 1 describes the pseudocode of our change detection algorithm and Figure 3.2 illustrates the different steps in the algorithm: (i) approximate entropy computed over a sliding window, (ii) canny edge detection, and (iii) removing nearby edges for a washing machine power trace. Given the change points, our technique then partitions each active period into segments, where each segment represents the power usage observed in a specific active state.

Algorithm 1 Changepoint detection to mark active state changes using Approximate Entropy

1: **procedure** ENTROPY-CHANGEPOINT(X, ϕ, δ)
2: **Initialize:** $H \leftarrow []$
3: $H.append(ApEn(X[i : i + \phi])) \quad \forall i \in [1..|X|] - \phi]$
4: $\varepsilon_{all} \leftarrow CannyEdge1D(H)$
5: $\varepsilon \leftarrow RemoveCloseEdges(\varepsilon_{all}, \delta)$
6: **return** ε

[1] The ApEn computation requires us to set two additional parameters (sequence length, set to $M = \phi/4$ and filtering level, set to $R = .2 \cdot \sigma(X)$) that are not shown in the pseudocode.

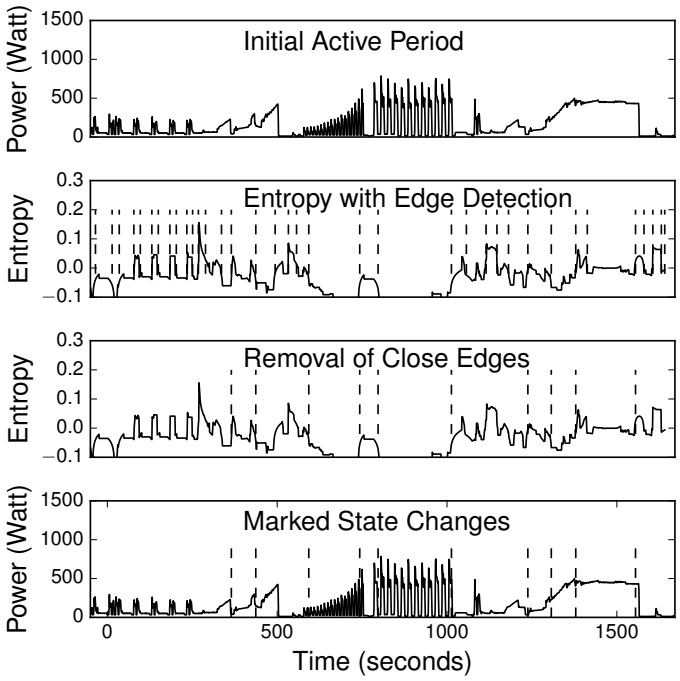


Figure 3.2. Approximate Entropy based change detection using canny edge detection on a washing machine trace

- **Step 3: Cycle Detection -** Certain loads may transition through repeating cycles of active states, yielding cyclic behavior that manifests itself as repeating patterns of observed power usage. Hence, rather than modeling the load as a linear sequence of active states, we search for repeating sub-sequences of active states that represent cyclic behavior within each active period or repeating patterns within an active state. We use *autocorrelation*, a standard time series technique, to discover repeating power patterns within an active period. The autocorrelation of a periodic signal will exhibit a *local maxima* at the time multiples of the original signal's underlying period. Thus, we compute the autocorrelation of the active period time series for different lag values to determine cycles. To illustrate this process, we choose a portion of the washing machine trace and show the corresponding autocorrelation values for the identified cycles (see Figure 3.3).

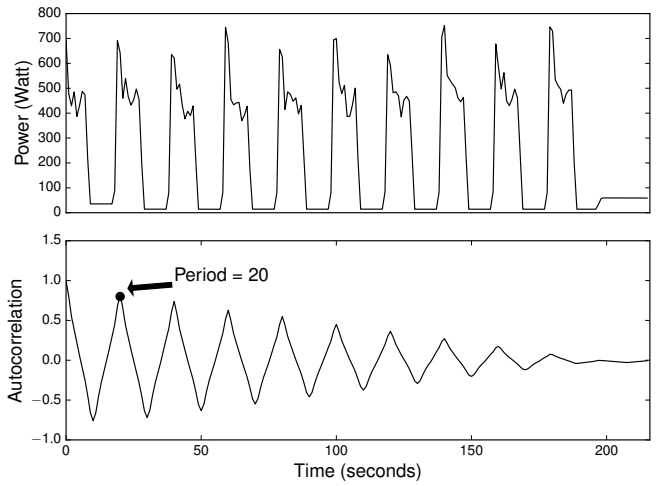


Figure 3.3. Autocorrelation plot of a time segement in an active duration of a device

- **Step 4: Model Fitting -** After extracting a time series segment for each active state within an active period, our technique then turns to the key problem of deriving an analytic model that describes the power usage variations observed within each state. Recall from the previous section that a basic load can exhibit on-off, on-off decay, on-off growth, stable-min or stable-max behavior, depending on whether it is resistive, inductive or non-linear. We use analytic closed form equations to capture the behavior of the first three types of loads and use probability distributions to capture the behavior exhibited by latter two types of non-linear loads. The model fitting process involves fitting a curve onto the time series data for the first three load types and fitting a distribution onto the data for the latter two. Since we have no a priori knowledge of the load type, our approach tries to fit different types of curves or distribution and chooses the best fit.

  First, to determine whether to fit a curve or a distribution, our technique determines if there are noticeable trends in the data (i.e., on-off, on-off decay or growth) or if it is derived from a random process (stable min or stable max). This is achieved by differentiating the time series of the load and observing the change in standard deviation. Our insight is that the standard deviation of the differentiated time series should

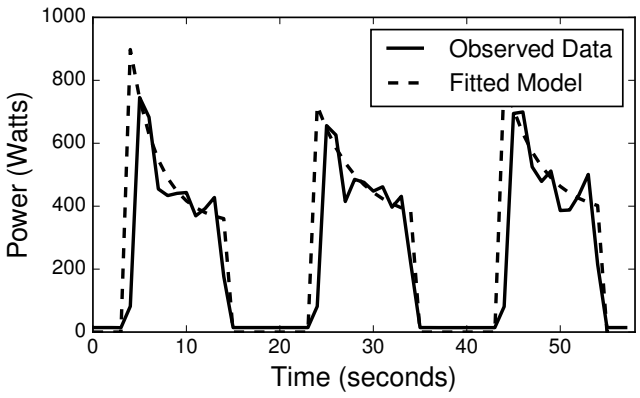


Figure 3.4. *On-Off Decay* model fit on a time segement in a active duration of a device

decrease for trending data and increase for data derived from a random process. This step enables our technique to determine whether to fit a curve or fit a distribution for each time series segment corresponding to an active state. In the former case, our technique then attempts to fit a linear segment, an exponential decay curve and logarithmic growth curve onto the data using *non-linear least squares* method. In the latter case, our technique attempts to fit both the gamma and the log-gamma distributions onto the data using the *Maximum Likelihood Estimation* (MLE) method. In either case, we choose the curve or the distribution that is the best fit in terms of explaining the observed data. Specifically we use *goodness-of-fit* measures, discussed later to choose the best fit. The output of this step is a classification of each active state as a particular type of base load and the parameters of the derived model (i.e., curve or distribution) for that base load. Figure 3.4 illustrates the *on-off decay* fit on the part of the time segment shown in Figure 3.3.

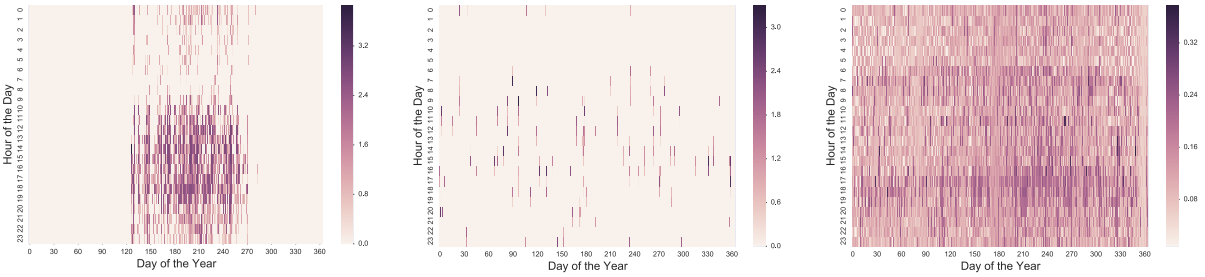
- **Step 5: Device Model Generation -** The previous step derives a unique model for each non-repeating active state within an active period and repeats this process for each active period in the raw time series. This yields a collection of models and our final step derives an overall device model from this collection of base models. This is achieved by creating a multi-tuple record comprising models for each active period.

Peak no.	P_{stable}	P_{peak}	λ	timelength
1	339.78	897.85	0.33	12
2	342.68	719.03	0.22	12
3	366.87	805.28	0.25	12
Mean	349.77	807.38	0.27	12

Table 3.1. Variation in parameters of the active duration of a device shown as a frequency table with mean value.

Each tuple contains information on the state number (in a given active period), period (or 0 if no period is found), the chosen label for the model (on-off decay, stable max etc.), the fit parameters (P_{stable}, P_{peak}, λ and time length for on-off decay), and overall segment length. For the segment shown in Figure 3.3, for instance, the tuple $\langle$Segment Number, Period, Model, Fit Parameters, Segment length $\rangle$ is given by - $\langle 11, 20, On - off \quad Decay, *params, 200 \rangle$.

In the case of cycles within an active period, the same basic model will be found repeatedly. However, due to the power behavior of electrical loads, there may be slight differences in the observed power values or patterns for different observed instances of the same state. Hence, the computed parameters of the load will vary slightly from one instance of the state to another. Our overall model can capture the variations at different degrees of accuracy. A more accurate description is less compact but captures the observed variations more faithfully. Conversely, a more compact model is less accurate and also more approximate. Currently, our technique supports three representations for capturing parameter variations across repeating instances of the same active state: (i) a single mean value for each parameter across all instances (the most concise representation, but also the least precise), (ii) a frequency table, or (iii) a probability density as multiple dimensions in a parameter hyperspace (the most precise). Table 3.1 displays the frequency table and the mean value for all the parameters for the time segment shown in Figure 3.3. Note that a single mean value for each model parameter will lose subtle variations exhibited by the load, while a probability



(a) Airconditioner (b) Dryer (c) Refrigerator

Figure 3.5. Energy consumption of devices in kWh over different time of the day and day of the year

density captures the likelihood of all the possible values of the different parameters of a model.

3.3.3 Usage Modeling

The usage model captures how a load is used within a certain environment by its users. Regardless of whether the load is a foreground load or a background load, the usage of a load is captured by how frequently it activates and when. Hence, the usage patterns can be captured by three parameters: (i) start time, (ii) length of an active period, and (iii) length of an inactive period. Note that the three parameters are not independent—the end of an inactive period defines the start time of the next active period. Nevertheless deriving all three parameters enables us to capture both the frequency of usage as well as seasonal dependencies (e.g., load is only active in the evenings, or only on weekends, or only in the summer etc). Figure 3.5 shows energy consumption (in kWh) in the form of a heat map for an AC, a clothes dryer and a refrigerator for each hour of the day for an entire year. The figure shows that the AC is used predominantly in the summer, while the refrigerator is active multiple times every single day on account of being an ”always-on” load. The dryer is typically used only once or twice a week.

- **Step 1:** To capture various usage patterns, our technique first determines the smallest time window (e.g., day, week, month or year) over which the load exhibits statisti-

cally significant usage variations. We start with the largest time window present in the trace (e.g., a year or a month) and compute the frequency distribution of start times over this time window. We then compute the *coefficient of variation* i.e. mean normalized standard deviation, for the start time frequency ν. We then recursively proceed to the next smaller time window (e.g., pick a week if the previous window was a month) and repeat the process of computing the frequency distribution of start times over this window and the coefficient of variation (COV) until the COV is found to be greater than 1. Thus, we pick the smallest time window (i.e., the most compact temporal representation) to model usage while ensuring that we do not miss any statistically significant variations in usage of the load.

- **Step 2:** Given the appropriate time window over which usage should be modeled, our technique then uses the start times and lengths of active and inactive periods extracted from the power series trace to compute (i) a histogram of start times over the time window, and (ii) histograms of active and inactive period lengths. We then use the Kernel Density Estimation (KDE), a non-parametric method for data smoothing, to compute a probability distribution over each histogram. This process yields three probability distribution functions for the start times, active and inactive period lengths, respectively. The joint probability distribution function over these three parameters represents the usage model for the load.

3.4 NIMD Implementation

We implemented a prototype of our NIMD algorithm in python using the SciPy stack. SciPy stack has a collection of powerful scientific computing libraries for data processing. Our prototype takes a raw power trace as input and outputs a device model and the usage model for it using techniques described in the previous section. The overall model fitting component and Kernel Density Estimation uses specific modules from the SciPy library. For calculating Approximate Entropy, we used PyEEG [16], an open source python mod-

Name	#Devices	Duration	Frequency	Region
AMPds	24	2 years	1 Minute	Canada
Smart*	26	3 months	1 second	USA
Tracebase	158	few days	1 second	Germany

Table 3.2. Datasets used for evaluation

ule for data processing for EEG data. For other statistical mechanisms, we used standard python libraries. The model derived from the trace can then be employed for a number of higher level energy algorithms. In addition, the model, which is a compact description of the device, can be also used to create a synthetic traces that "faithfully" mimic the load's actual power behavior, as discussed next.

3.5 Evaluation Setting

3.5.1 Datasets:

We used device-level electrical data from three publicly available datasets: AMPds [72], Smart* [19], and Tracebase [86]. Table 3.2 describes the key characteristics of these datasets. AMDds is the smallest of the three datasets, but has load data for two years. Tracebase is the most extensive dataset in terms of number of loads, while the Smart* has appliance-level data at a 1-second resolution over a period of 3 months.

3.5.2 Metrics:

To analytically evaluate the goodness of fit for *on-off*, *on-off decay* or *on-off growth* models described earlier, we use Mean Absolute Percentage Error (MAPE), a standard statistical measure of accuracy expressed as a percentage value. The formula for calculating MAPE for a given device with power consumption data represented as $X^{data}_{[1..k]}$ and the fitted model $X^{fit}_{[1..k]}$ is given below.

$$MAPE = \frac{100}{n} \cdot \sum_{k=1}^{K} \left| \frac{X_k^{data} - X_k^{fit}}{Mean(X_{[1..K]}^{data})} \right| \tag{3.5}$$

For *stable min* and *stable max* models, we use Kullback-Leibler (KL) divergence, a measure of the difference between two probability distributions. KL divergence of probability distribution Q from P is symbolized as $D_{KL}(P||Q)$. However, KL divergence is not a metric as it is not symmetric. In practice, probability P is the distribution of the data and Q is the proposed approximation for P. In our case, Q is the $Gamma$ distribution for *stable min* and the $LogGamma$ distribution for *stable max*. The lower the MAPE or KL divergence values, better is the fit.

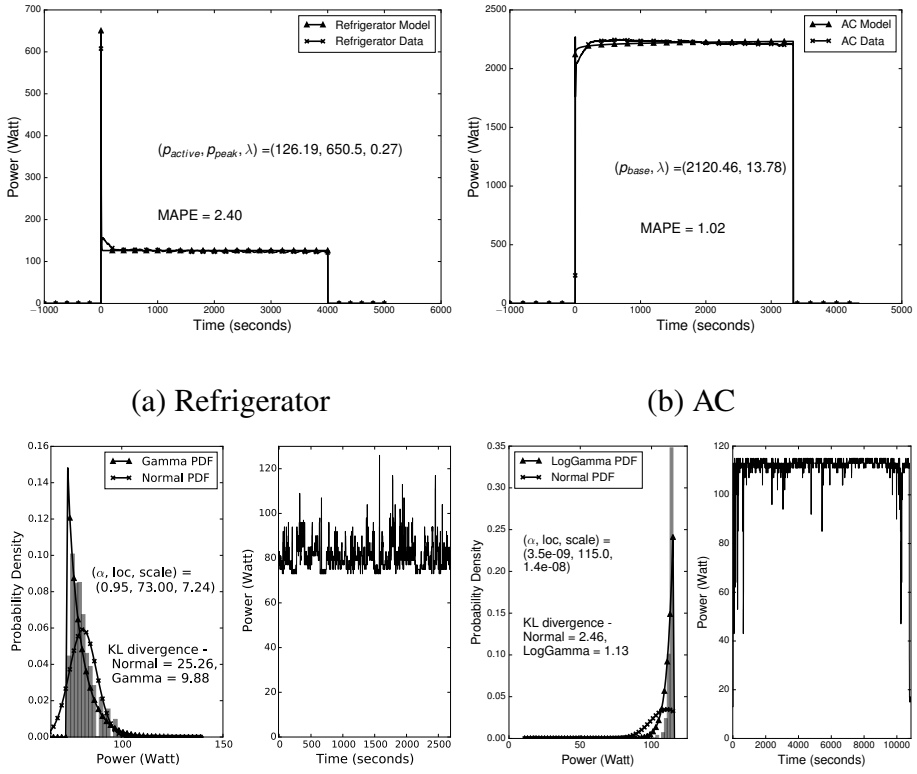
$$D_{KL}(P||Q) = \sum_i P(i) \cdot \log \frac{P(i)}{Q(i)} \tag{3.6}$$

3.6 Evaluation

In this section, we evaluate the efficacy of our NIMD approach for device and usage modeling.

3.6.1 Device Modeling of Basic Loads

Figure 3.6 illustrates the performance of model fit on the 4 appliances from the Smart* dataset. These appliances are - (a) a Refrigerator, (b) an AC, (c) a CRT-Monitor, and (d) a LCD-TV fitted with *on-off decay*, *on-off growth*, *stable min*, and *stable max* models respectively. The learnt model parameters are also shown in each figure. Figure 3.6 (a) and (b) show the MAPE values for the fitted model on the data. For the two examples shown for curve fits in Figure 3.6(a) and (b), we get a MAPE (error) of 2.4% and 1.02%. Figure 3.6 (c) and (d) show the KL divergence of data from $Gamma$ and $LogGamma$ distribution respectively. These figures also show the KL divergence of data for a baseline $Normal$ distribution fit. Intuitively, KL divergence is the penalty on compressing data to be represented as the proposed distribution. The figure shows that KL divergence of the our



(a) Refrigerator (b) AC

(c) CRT-Monitor (d) LCD-TV

Figure 3.6. Basic Load Models for *On-off Decay*, *On-off Growth*, *Stable min*, and *Stable max* with fitted models

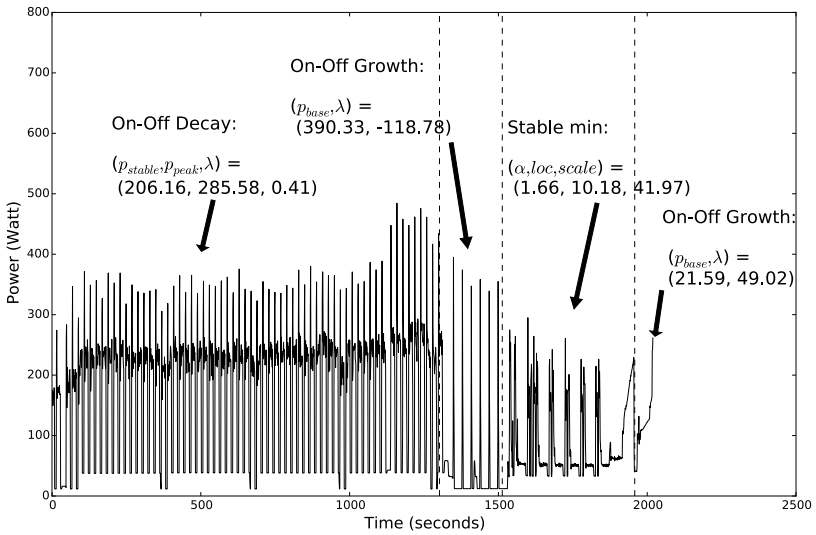
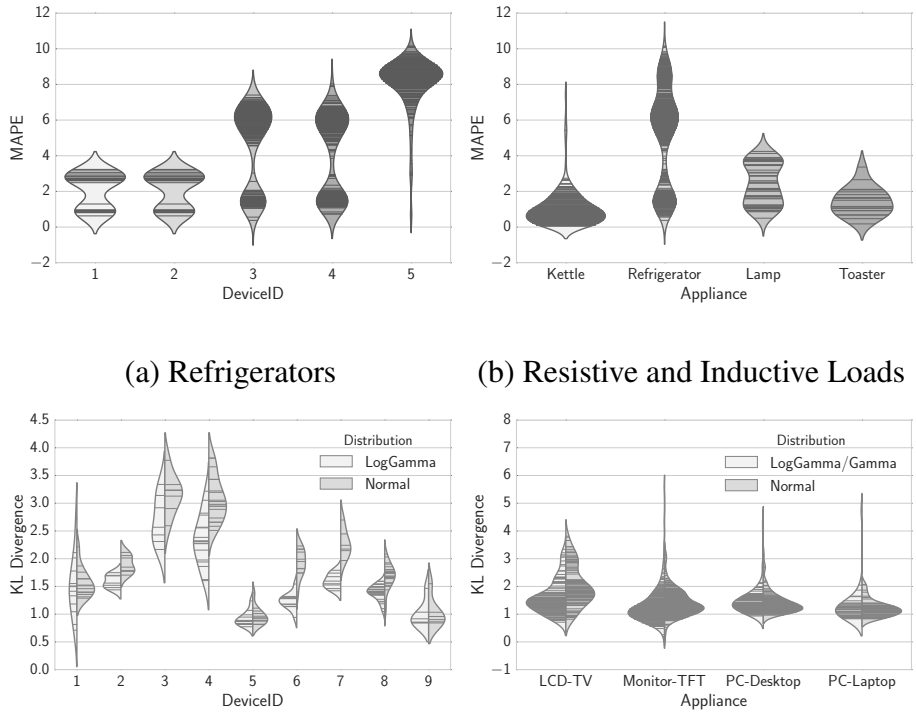


Figure 3.7. Model learnt for a composite load

proposed $Gamma$ and $LogGamma$ distributions is a more than a factor of 2 lower than the baseline $Normal$ distribution. Finally, Figure 3.7 shows the overall model learnt for a washing machine, a composite load.

3.6.2 Accuracy of the device models:

To evaluate the accuracy of model fit, we ran it on a number of appliance loads of various types in the tracebase dataset. In Figure 3.8(a), we have a violin plot showing the MAPE values for 5 refrigerators over curve fit on several active periods of the device. The horizontal stick in these plots represents each underlying datapoint corresponding to a measurement for an active period. The thickness of the graphs for different devices corresponding to the MAPE values on the y-axis is indicative of the frequency distribution of the datapoints. Overall, more than 1000 active periods spread across 5 devices are shown in this figure. MAPE values for 2 of the refrigerators are almost below 3%, whereas it is between 1-7% for 2 other refrigerators. For one refrigerator, we found a comparitively much poorer fit in the range of 6-10%. Figure 3.8(b) shows an appliance type-wise view of the error in the curve fits for 4 inductive (Refrigerator) or resistive (Kettle, Lamp, Toaster)



(a) Refrigerators (b) Resistive and Inductive Loads

(c) LCD-TVs (d) Non-linear Loads

Figure 3.8. Goodness of Fit measures for different appliances from Tracebase dataset

load types. The graph represents more than 1300 active period data for 7 Lamps, 6 Kettles and 2 Toasters along with the 5 refrigerators shown in (a). As shown, the resistive loads have MAPE values below 4%.

As discussed earlier, for distribution fit we use a relative measure called KL divergence. Here, again we compare our proposed distributions to a baseline $Normal$ distribution. Figure 3.8(c) shows the violin plots for more than 200 active periods spread across 8 LCD-TVs. For 5 devices the KL divergence improves by modeling the active period traces as a $LogGamma$ distribution by a factor of 1.5. For the other 3 devices, there is no appreciable difference between the two distributions. Figure 3.8(d) represents appliance type-wise spread of KL divergence for non-linear loads such as TFT-Monitor, Desktop-PC, and Laptop along with LCD-TVs representing more than 1000 active periods. Except for LCD-TVs, we do not find any improvement (or worsening) in KL divergence by model fitting proposed one-tailed distributions over $Normal$ distribution.

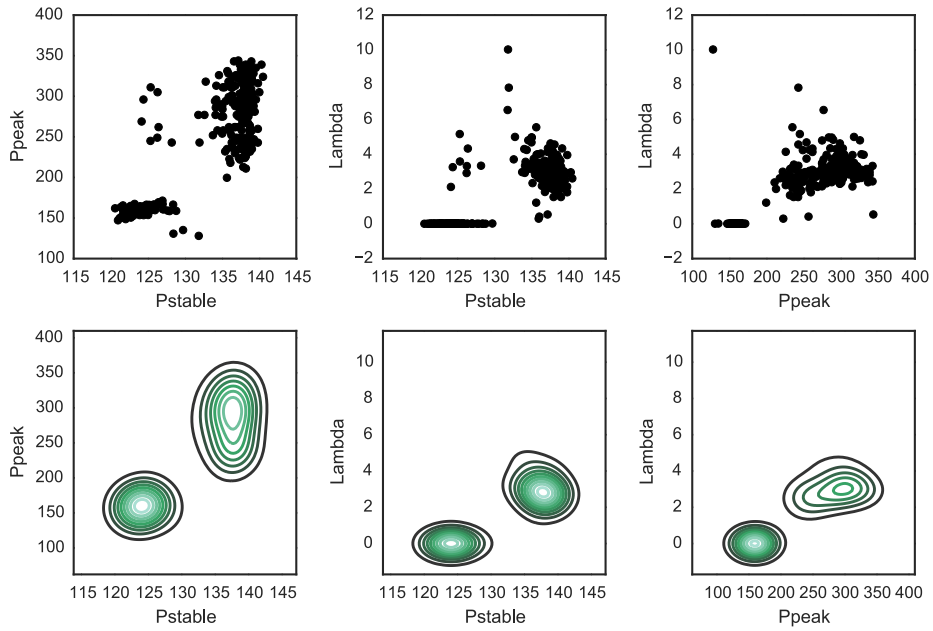


Figure 3.9. Variation in parameters over several active periods

3.6.3 Descriptiveness of the Model:

The model parameters of an electrical load are not static and the variation in them must be captured for building a realistic model. Figure 3.9 shows the probability density over the 3 dimensions of the parameter space (p_{peak}, p_{stable}, and λ) obtained from applying NIMD algorithm on the different active periods of a refrigerator from the TraceBase dataset. We observe that the 3 parameters vary from one active period to the other. Figure 3.9 illustrates how a probability density is more precise than a frequency distribution table (shown as a scatter plot) as it provides a smooth parameter space with just a few data samples.

3.6.4 Usage Modeling

To evaluate the efficacy of the usage modeling, we used loads from the AMPds dataset since it contains consumption data for a period of 2 years. With an adequate amount of data, we can choose the smallest time window which captures the usage variations of a device. Earlier, we discussed how the joint probability distribution of start times over an optimal window and the length of the active and inactive periods capture the usage model

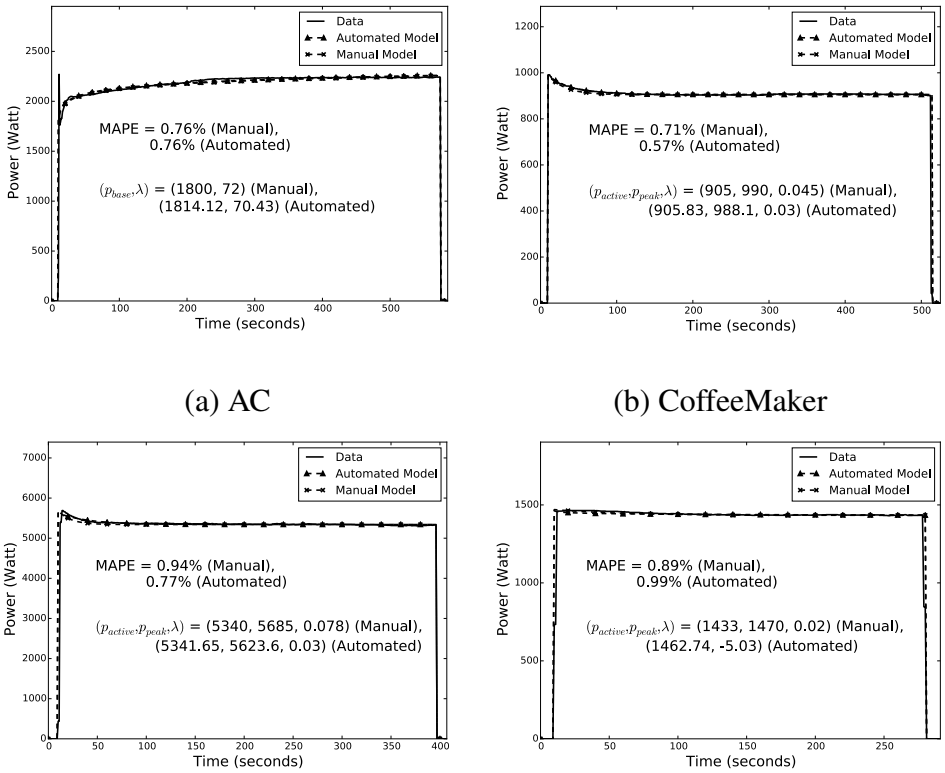
Device (Window)	Start times/interval		Active length		Inactive length	
	μ	σ	μ	σ	μ	σ
Dryer(W)	4.2	2.2	41.8	11.7	1891	2220
Washer(W)	5.2	2.7	50.9	24.3	1557.1	2114.7
Dishwasher(W)	3.7	1.3	75.8	39.6	2034	2160
Fridge(D)	39.6	4.0	13.5	9.8	22.5	9.7
TV(D)	1.9	1.0	67.7	47.9	522.8	501.5
WOE(M)	3.6	1.8	105.9	399.6	8978	7901

Table 3.3. Usage Patterns of devices with Daily (D), Weekly (W) or Monthly (M) window

of any device. Since the joint probability of these 3 variables is difficult to plot, we use Table 3.3 to show the mean and the standard deviation of a number of active periods in a time window (optimally selected from the data) with the length of the active and inactive periods. The time window selected for the different devices matches the intuitive values that would have been manually selected for the different devices. For example, our models indicate that the approximate duty cycle for the refrigerator is around 36 minutes (average active + inactive period lengths). Our usage models also capture that devices such as Dryers, Washing machines and Dishwashers are used 3 to 6 times per week.

3.6.5 Automated versus Manual Modeling

To compare our automated approach with models manually derived by experts, we obtained load data and manually derived models reported in [18] from the authors. We used NIDM to derive models for the loads and then compare NIMD's models with the manually derived ones. Figure 3.10 shows a comparison between the manual and the automated modeling approaches. We were able to classify each of these 4 appliances with the correct basic load type. Further, the learnt parameters were very close to the manual modeling



(a) AC (b) CoffeeMaker

(c) Dryer (d) Toaster

Figure 3.10. Comparison of automated v/s manual modeling

values shown in [18]. The error associated with both manual and automated modeling was lesser than 1% in all 4 cases. Thus, our automated approach derives models comparable to human-derived fitted models using domain knowledge (e.g., load type).

3.6.6 Case study: Synthetic Trace Generation

While our models can be used for many energy management tasks, they can also be used to derive a synthetic power trace that is statistically similar to the original trace. For this, we need to sample the usage distributions to compute start times of each active and inactive durations. To derive the parameters such that the synthetic trace mimic the original load usage, we need to draw samples from joint probability distribution computed by the usage model. To do so, we employ a state-of-the-art Markov Chain Monte Carlo sampling method called the *Metropolis-Hastings Algorithm* [27] that generates a sequence of samples

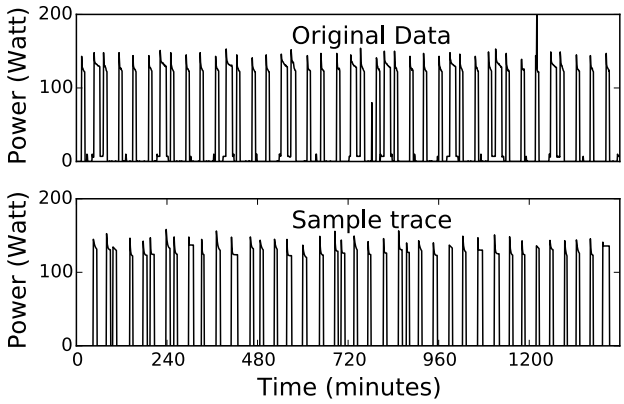


Figure 3.11. Synthetic trace generated using out models.

through a *random walk* over the sample space. Once the start time is computed, the usage model, which is itself a sequence of analytic models for each active state, is then used to create a power trace for that active period. In the case of non-linear loads, where the device models are distributions, we sample the distribution to create a power trace. At the end of an active period, we set the power to the standby power level for the inactive period. The process repeats for the next start period.

Figure 3.11 shows the original load trace and the sample trace generated synthetically from the first sample taken from real data after 10^4 iterations. By ”replaying” the usage and device models, we observe that the synthetic trace exhibits similar power usage as the original load trace.

3.7 Related Work

Due to the large-scale deployment of smart meters by utilities, there has been a resurgence in interest in energy analytics techniques, such as NILM, in both academia [12,50,66] and industry [24]. NILM-based energy analytics have been used in different scenarios, such as opportunistic load scheduling for capping peak demand [20], learning thermostats schedule [56], etc. However, prior work on NILM generally uses simple *on-off* models for electrical loads, which, as we show, are highly inaccurate. Thus, an important challenge

is the ability to analytically model the behavior of a variety of residential loads. Earlier work [17, 18] has demonstrated that most appliances map onto few basic types that exhibit a compact set of features. This prior work shows how to manually construct models for the basic load types, but does not show how to automatically derive models, especially for complex loads that are time-consuming to manually model.

In this work, we propose an algorithm to automatically derive a model for each appliance from its empirical measurements. Our technique is analogous to disaggregation where an energy usage trace of a compound load is automatically disaggregated into a set of basic load types and the parameters of each basic load type are automatically learned. Further, we also model the interaction of devices with residents to build a usage model for them. Finally, our NIMD techniques adapt and extend multiple methods from probability, statistics, and information theory to the energy analytics domain. These methods provide a strong theoretical framework for automatically deriving models of electrical load behavior.

3.8 Conclusions

In this chapter, we presented a new approach for automated unsupervised derivation of the device and usage models of residential loads. We presented our NIMD approach that uses concepts from power systems, statistics, and machine learning to automate loads modeling. Our experimental evaluation showed that our automated models are within 1% of the ground truth and very close to those derived manually by experts and yield good fits for a range of loads. A current limitation of our approach is that they only handle sequential composite loads, where the base loads activate in sequence, and do not handle parallel composite loads. As future work, we will study methods that combine NILM disaggregation with our NIMD approach to handling parallel composite loads.

CHAPTER 4

BLACK-BOX SOLAR PREDICTOR FOR SMART HOMES

This chapter presents SolarCast, an algorithm to automate modeling of residential rooftop solar installations. We initially start with describing our automated black-box model generation that produces site-specific solar panel model. We conclude with a detailed evaluation illustrating the performance of our algorithm along with a case-study.

4.1 Motivation

Solar power predictions can play a crucial role in reducing the need for expensive storage capacity while also reducing the amount of energy fed to the grid by using it by locally to run electrical loads in a household. However, for residential rooftop installations, it is often difficult to obtain the panel properties and installation-specific configuration parameters because homeowners do not know these technical details. Further, it is even more challenging to know dynamic factors, such as shade, foliage, and pollen, which vary either seasonally or irregularly. Consequently, designing prediction models for residential rooftop installations requires a new approach that should automatically learn panel properties and configuration parameters with limited historical power data. Additionally, these models should automatically adapt to the dynamic parameters that vary over time, such as tree shade and snow, dust, and pollen. Table 1 shows static and dynamic panel properties.

Thus, to generate solar power prediction models at scale, for any solar installation in the country, one must design a black box model that only requires a site's location and minimal historical generation data to generate a customized prediction model, tailored to that particular site.

Parameters	*Variability*	*Our Mechanism*
Panel Parameters	Static	ML Regression
Tilt	Static	Optimization Parameter
Orientation	Static	Optimization Parameter
Temperature Coefficient	Static	Optimization Parameter
Tree Shade	Dynamic	Adaptive Learning
Snow/Dust/Pollen	Dynamic	Adaptive Learning

Table 4.1. SolarCast employs different techniques to capture panel and configuration parameters.

4.2 Automated Black-box Model Generation

While there has been significant work in predicting solar generation, our approach differs from prior work in three significant respects. First, SolarCast automates the model generation process—it requires minimal initial inputs from the user and requires no manual intervention by the user to generate a model. Second, SolarCast uses a black-box modeling approach to learning the value of unspecified parameters. Third, SolarCast continuously retrains and refines the model using live data, while also using live data to adjust for the impact of dynamic site-specific factors that are impossible to learn.

Similar to many approaches presented in the prior work for solar predictions, SolarCast also employs machine learning techniques to derive its model. However, the primary difference from prior work is that SolarCast *automates the process of learning the model* itself, which we refer to as automated model generation. Such an automated model generation approach is key to scaling SolarCast to large numbers of small-sized deployments. At the heart of SolarCast's automated model generation is a black-box modeling approach, which represents another departure from prior work that typically uses white box techniques. To better understand the differences between the two, consider the following canonical white box modeling approach based on machine learning. The solar deployment is a "white box," which means that all important parameters of the deployment, such as its panel type, tilt, orientation, efficiency, etc., are assumed to be known. Further, a history of past generation

data at different times of the day and seasons of the year is given, along with the observed weather conditions at those times. The machine learning approach then simply learns a map between the specified inputs and the observed solar output.

The learnt model is a "function" that, given certain inputs, such as weather and time of day, will compute the expected solar output under those conditions—based on the model's correlations. Much of the prior work, including some of our own [94], take such an approach. In contrast, in a black box approach, the solar deployment is assumed be a "black box" where site specific parameters, such as the number and type of panels, tilt, orientation, shadows, etc., are all *unknown*. Instead, a past history of weather data and the observed solar generation (inputs and outputs of the black box) are given and all unknown parameters must be learned. Intuitively, this is done by searching for the combination of these unknown parameters that best explains observed outputs. Some dynamic parameters that are challenging to learn are accounted for by adjusting the predictions dynamically. In this manner, a black box method is more complex than white box methods, but also requires fewer inputs and less training data (since the models are continually refined as more live data becomes available).

4.2.1 Accuracy Metric

Note that we use the Mean Absolute Percentage Error (MAPE) as our preferred statistical metric to measure a model's accuracy. Though the Root Mean Squared Error (RMSE) is a well-known statistical metric, we prefer the MAPE because it is capacity agnostic, which allows us to directly compare accuracy across panels with widely different capacities. Such comparisons are not possible with RMSE. Further, since weather forecasts are occasionally erroneous, the MAPE is less sensitive than the RMSE to occasional large errors. Thus, MAPE is a better metric to illustrate the forecast prediction error. The MAPE for n samples is expressed as:

$$MAPE = \frac{100}{n} \cdot \sum_{t=1}^{n} \left| \frac{A_t - P_t}{A_t} \right| \tag{4.1}$$

Here, A_t and P_t are the instantaneous actual and the predicted generation value at time t. Unfortunately, problems can occur when calculating the MAPE using a series of small denominators, as they will substantially increase the MAPE. Instantaneous generation can vary significantly, briefly dropping to low values even during near ideal conditions. To circumvent this 'divide by zero' problem, we change the denominator in the original formula from A_t to A', which is the average value over the entire time interval t. Further, MAPE values calculated this way are insensitive to the inclusion of nighttime actual and predicted values as both would be zero. We use the below formula to report prediction errors.

$$MAPE = \frac{100}{n} \cdot \sum_{t=1}^{n} \left| \frac{A_t - P_t}{A'} \right| \tag{4.2}$$

Below, we start our discussion with the description of the process of model generation for predicting solar power. This essentially outlines the various inputs to the function of the learned model. After which, we move on to describe two distinct machine learning techniques to build our custom site-specific model to predict solar power while learning the hyperparameters. First, we present a constrained least-squares curve-fit method, which uses training data to learn hyperparameters and the linear combination of features presented in the previous section. Then, we present the deep neural network architecture which is capable of learning complex relationship that exists between weather parameters and the solar power. Following this, we show how our models can generate a site-specific model for a solar installation in an online setting where fresh solar generation data is available to continuously refine our models with diminishing prediction error.

To generate our model, we first prepare a forecast $\rightarrow$ power model that predicts solar power from weather forecasts for a sun-tracking solar installation that always keeps its panels facing the sun, e.g., oriented towards the equator. Next, we extend the model to automatically learn the static configuration parameters, such as tilt and orientation. As

photovoltaic (PV) panels use semiconductor-based P-N junctions, an increased temperature also increases the resistance and thus reduces the overall current (and power) generated. However, different PV panels have a different tolerance to high temperatures. For example, amorphous silicon solar cells are more resilient than mono and polycrystalline cells. Thus, we also need to learn an adjustment factor for a given installation for different ambient temperature changes. We then further extend the model to create an adaptive version that not only automatically learns the configuration parameters, but also accounts for the dynamic environmental factors, such as snow, dust, and pollen.

4.2.2 Solar Model

Much of the prior work focuses on predicting solar irradiance from weather forecasts and then using the irradiance $\rightarrow$ power formula to predict the solar power. Since designing both the forecast $\rightarrow$ irradiance model and irradiance $\rightarrow$ power model require historically observed irradiance, this approach is not scalable to millions of rooftop installations because the historical irradiance data is generally not available for these sites. So, instead, we focus on designing a prediction model that directly predicts solar power from weather forecasts for any solar installation at any location on earth. We first assume the optimal configuration for the solar panel, i.e., the panel is always facing the Sun and is normal to the solar radiation.

As described in prior work [94], weather metrics exhibit complex relationships with solar intensity, which can be captured by advanced techniques, such as high-dimensional machine learning regressions. Feature selection and feature engineering play an important role in machine learning. Similar to prior work [94], we consider all weather parameters included in National Weather Service forecasts, including sky cover, temperature, humidity, dew point, precipitation potential, and wind speed, as input parameters, but unlike prior work, we create a new feature set by normalizing all weather parameters by the cloudless irradiance (e.g., by multiplying by the cloudless irradiance). This normalization of features

is based on our intuition that the same weather parameter always affects the solar irradiance in the same proportion, regardless of the time. *For example, if at 6pm a certain weather parameter causes cloudless irradiance to be cut in half, then if the weather parameter is the same at 12pm, it will also cause the cloudless irradiance to be cut in half (even though cloudless irradiance at 12pm is different than at 6pm).* Additionally, since the cloudless irradiance depends on the altitude and azimuth angles of the sun, by multiplying the cloudless irradiance by the weather parameters our model also captures the seasonal and diurnal variations of the sun's position. Consequently, we formulate the power prediction regression model as:

$$P_t = f\left(S_t^{cloudless} \cdot W_t\right) \tag{4.3}$$

Here, P_t and $S_t^{cloudless}$ are predicted power and cloudless irradiance, respectively, at time t, W_t is a vector of size i containing the forecast of the weather parameter at time t, and f is the function that we determine using machine learning regression. This novel feature engineering enables prediction of solar power at any time of the day without learning a separate model for different hours, a drawback of the proposed model by [94]. Our insight and the normalization above is the key to enabling our techniques to use all historical generation data as training data for the same model. Prior work has had to learn different models for different time periods, e.g., one model per month, since the maximum solar generation varies throughout the year. These prior approaches require more time to collect training data, e.g., multiple years, and are significantly less accurate, as even the maximum solar capacity each day within a given month will vary.

4.2.3 Black-box Learning of Static Parameters

The above model assumes the optimal configuration of solar panels and might work well for a sun-tracking solar installation. However, a typical rooftop installation does not have the optimal configuration, since its configuration is dictated by the tilt and orientation

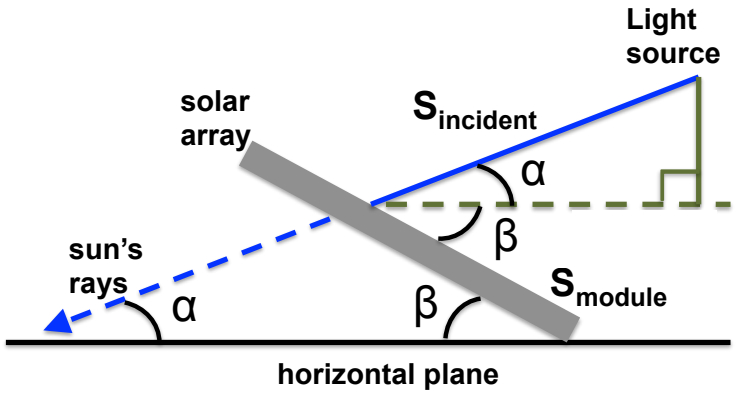


Figure 4.1. Tilt of panel (β) and solar elevation (α)

of the roof. Further, panel properties and configuration parameters vary widely across different sites and are often unknown to owners. Thus, to automatically learn these static parameters and generate site-specific prediction models we modify the above regression model as follows:

$$P_t = f'\left(S_t^{module} \cdot \nu_t \cdot W_t\right) \tag{4.4}$$

P_t and W_t are the same as in the previous section. In addition, S_t^{module} is the perpendicular component of the cloudless irradiance on the panel, ν_t is the adjustment factor due to the changes in the ambient temperature, all at time t. We then learn the function f', for given values of static parameters, using the machine learning regression technique from the previous section. Since the component of the cloudless irradiance perpendicular to the surface of the panel depends on - (i) the panel's tilt and orientation, and (ii) the Sun's altitude and azimuth, S_t^{module}, is expressed as:

$$S_t^{module} = S_t^{cloudless} \cdot (\cos\alpha_t \sin\beta \cos(\psi - \theta_t) + \sin\alpha_t \cos\beta) \tag{4.5}$$

Here, α_t and θ_t are the altitude and azimuth of the Sun at time t, respectively, whereas β and ψ are the tilt and orientation of the panel, respectively (see Figure 4.1). The details involved in deriving this equation can be found in [95]. For simplicity, we assume the panel's

orientation to be same as the sun's azimuth in the figure. In addition to tilt and orientation, ambient temperature has a direct bearing on the power output from a photovoltaic panel.

To account for the impact of temperature on the panel output, we must calculate ν_t for different time periods t. As noted earlier, different panels have a varying amount of resilience to increased temperature. This depends on the type of panel, e.g., monocrystalline, polycrystalline, amorphous, etc., and the distinct manufacturing processes followed by panel manufacturers. For our black-box model, we assume information, such as the type of panel and manufacturer information, is unavailable. However, based on different adjustment factors discussed in the literature, we have the following general equation to account for temperature changes with hyperparameters a and b:

$$\nu_t = a \cdot (T_t^{cell} - b) \tag{4.6}$$

We learn the hyperparameters, such as tilt (β), orientation (ψ) with a and b for temperature adjustment, using the parameter optimization techniques detailed below. In summary, SolarCast automatically learns static configuration parameters such as tilt, orientation, temperature adjustment factor, and generates a custom site-specific prediction model for any solar installation ranging from a single panel rooftop installation to a large solar farm; it only requires the location and historical power data from the site.

4.2.4 Constrained Least-Squares Curve-Fit

We apply the constrained least-squares curve-fit technique on a training dataset to determine the function f, i.e. a linear combination of different features discussed in the previous section. Least squares regression is a simple and commonly-used technique to estimate a value to be predicted from a set of variables. Here, we leverage this technique to predicting solar power from weather forecasts. This regression technique minimizes the sum of the squared differences between the observed solar power and the power predicted by a function approximation of forecast parameters. Using this technique, we initially use his-

torical solar power data (also referred as training data), to learn optimal coefficients for the different features. Essentially, as we have few unknowns, in the form of features and hyperparameters, limited training data (more than the number of features and hyperparameters) is sufficient to build a reasonable model. Next, with the help of these learned parameters, we can make future predictions when we are supplied with expected feature values.

- **Hyperparameter Learning:** While learning the function f, we apply constraints on tilt and orientation to reflect the realistic range of values, such as the tilt of a panel can only range from 0 ° to 90 °. Thus, apart from learning the coefficients for the combination of individual features, this technique can also learn the hyperparameters that best fit the training data. The learned hyperparameters with the coefficients together help in predicting solar power.

- **Model variation:** The function f, described in Equation 4.4, can be used to learn the static parameters and to predict the solar power. Hereinafter, use of constrained least-squares curve-fit on the function f defined by Equation 4.4 is referred as *Black box (Static)* model. However, apart from the static configuration parameters, dynamic environmental factors, such as tree shade, foliage, pollen, snow, and dust, also affect the power output. A few of these parameters, such as foliage and pollen, have seasonal variations, while others, such as leaves and dust, vary irregularly. Since, unlike large solar farms, rooftop installations are not cleaned regularly, we must account for the dynamic factors in our prediction model. Our intuition is that power output in the recent past contains some information about the impact of dynamic factors. To compensate for prediction errors due to the dynamic factors we add a new feature P_{t-24h}^{output}, which is essentially the power output at the same time the previous day, in the feature set. So, the predicted power at time t can be expressed as:

$$P_t = f'\Big(S_t^{module} \cdot \nu_t \cdot W_t, P_{t-24h}^{output}\Big) \tag{4.7}$$

Notations P_t, S_t^{module}, ν_t and W_t have been introduced earlier. Since P_{t-24h}^{output} is a dynamic parameter that changes every day, we can account for *surrounding characteristics* such as tree shade and *dynamic characteristics* such as snow, dust, pollen etc.

Hereinafter, use of constrained least-squares curve-fit on the function f defined by Equation 4.7 is referred as the *Black box (Adaptive)* model as it can adapt to account for dynamic environmental factors, that vary seasonally or irregularly, by automatically correcting the model to account for effects due to them.

4.2.5 Deep Neural Network with Custom Input Layer

A Neural Network is a model based on the human brain and nervous system. Similar to the biological model, the neural network model in machine learning consists of a network of neurons. Each neuron has multiple inherent parameters associated with each *neuron* that includes a weight term and a bias term. These terms are applied to one or more inputs received by the neurons. A *layer* of the neural network might contain one or more such neurons. Each layer has an activation function, which will fire the neurons in it depending on the values of the input along with the weight and the bias term. There are various activation functions which can be used in a neural network model. The weights and the biases for the different neurons in each layer are learned using *backpropogation* algorithm. This algorithm uses training data to tune these terms. The power of the neural networks lies in their ability learn arbitrary functions to map the inputs to the output. Further, one can use multiple layers of neurons stacked one top of the other with each of these activated using different activation functions. The resultant model renders even greater power to learn complex functions. These layered neural networks are called deep neural nets.

Deep neural nets have been applied in recent years to several domains, such as computer vision, natural language processing, and speech recognition. Theoretical advances in

training deep architectures with the advent of GPUs have now made deep neural nets the technique of choice for solving challenging problems in artificial intelligence.

In our case, we build a 4-layer deep neural network as shown in Figure 4.2 to learn the function defined by Equation 4.4. This deep neural network produces an expressive model that is capable of learning the complicated relationship between their inputs (features) and output (solar power). As shown in the figure, each layer, except the last, consists of an equal number of neurons. This number is equal to the number of features used in our model. The first layer receives the raw features discussed earlier, as inputs. The subsequent layers receive all the outputs of the previous layer as inputs. For the first two layers, we use a Rectified Linear Units (ReLU), the most popular activation function for deep neural networks. As we have formulated solar power prediction as a regression problem, we use a linear activation function for the next two layers. Clearly, as the solar power generated at a given time is a scalar, the last layer contains just one neuron that outputs the prediction. As with any machine learning model, its complexity needs to be controlled to avoid overfitting, which is typically caused by parameters taking extreme values. Deep neural nets have multiple neurons, each containing a weight and a bias term. These parameters could take large at the end of training and could perform poorly while making predictions. To avoid this, we use L2 regularizers on the layer parameters. Hereinafter, this model is referred as a *Black box (Deep)*.

We use a grid search technique to automatically find the tilt and orientation (hyperparameters) of a solar panel installation. The grid search algorithm finds the tilt and orientation that minimizes the MAPE value by cross-validating over training data. Firgure 4.3 shows the MAPE values for the different combination of tilt and orientation for a solar installation. Here, the tilt of 20 degrees and orientation of 180 degrees (panel facing south) has the lowest MAPE.

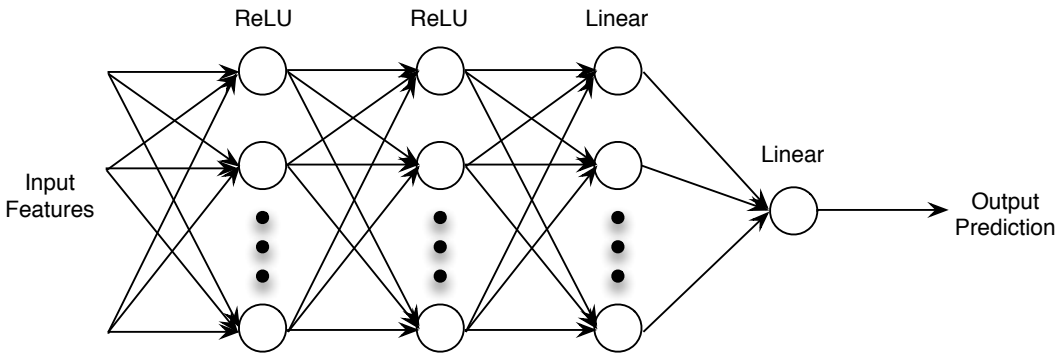


Figure 4.2. SolarCast's Deep Neural Network Architecture

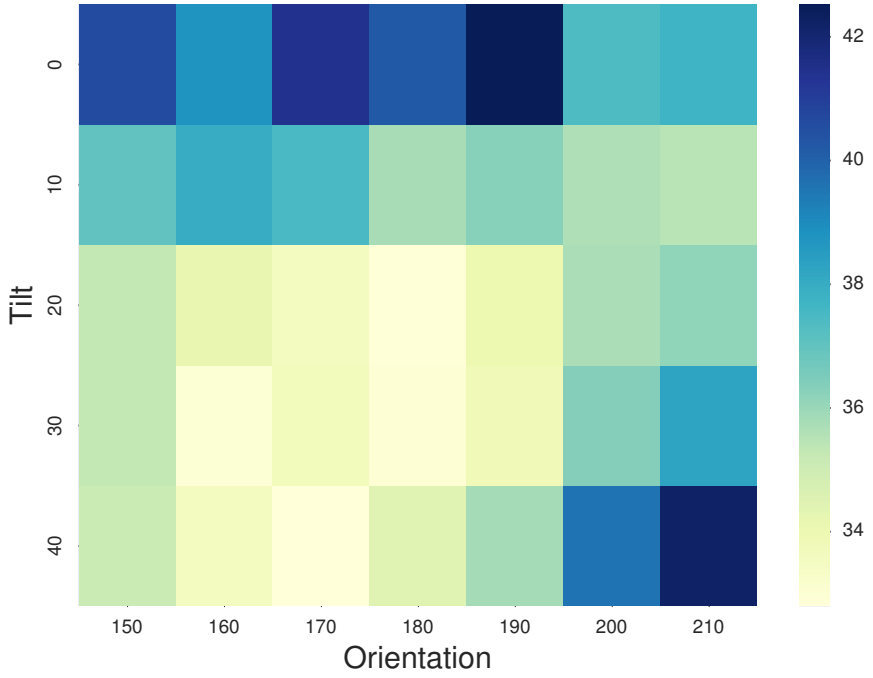


Figure 4.3. Grid Search over different values of Tilt and Orientation

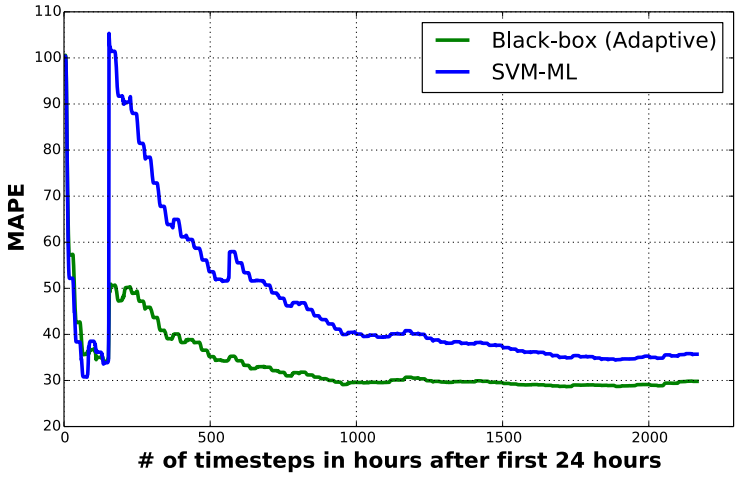


Figure 4.4. Prediction error for the online version of SolarCast's adaptive model and SVM-ML model.

4.2.6 Online Learning

Like any regression model, SolarCast also requires historical data (of several months) for training, which increases the barrier to using its services. Gathering sufficient historical data is especially challenging for new installations or existing installations that have not been continuously monitoring and archiving power generation since deployment. Further, not all homeowners install sophisticated monitoring devices, which can store data for several months or years. With limited training data, the machine learning models discussed earlier might generate poor results as with few examples the generalization error is higher.

To address issues with insufficient historical data, SolarCast employs an online algorithm that starts generating site-specific prediction models from as little as one to two days of historical data. Further, SolarCast stores the past data for each site and retrains (and refines) the model as it gets more recent data from the site. This step involves using the machine learning algorithms described earlier to be used repeatedly as and when new data is available. Albeit poor in the beginning, the error associated with the SolarCast's predictions will trend lower with more data available to correctly train the machine learning model.

To see how quickly our online approach achieves a prediction accuracy of the static or adaptive model generated with sufficiently large training data, we start the online model with just one day of training data. As shown in Figure 4.4, the online model rapidly converges, within 10%, to the static model within one month. This illustrates that any new or old installation can start using SolarCast service with just a few weeks of historical data. Further, we compare our online approach with the online version of the machine learning prediction model from [94] using a Support Vector Machine (SVM) with a linear kernel, hereinafter referred as the SVM-ML model. The figure shows that our adaptive model requires much less training data than the SVM-ML model to create site-specific prediction models. As SVM-ML learns a separate model for each time of the day, essentially leveraging only $(1/24)^{th}$ of the training data, the MAPE improves more slowly over time. Also, notice MAPE for both approaches stabilizes with more data. This graph uses the constrained least-squares curve-fit technique described below for learning the model. As we show in Section 5, the deep neural network model performs even better.

In summary, SolarCast's online learning technique can generate site-specific prediction models with as little as a few weeks of historical data and keeps on refining the models as and when it gets new data from the sites.

4.3 SolarCast Cloud Service

In this section, we first describe the high-level architecture of SolarCast, followed by our prototype implementation.

4.3.1 Architecture

SolarCast provides a web-based service (see *http://solarcast.cs.umass.edu*) that households can use to predict solar power generation from their own installations for short-to-medium timescales ranging from tens of minutes to a few days. Unlike prior services [83], which are either proprietary or require knowing installation- and panel-specific parameters,

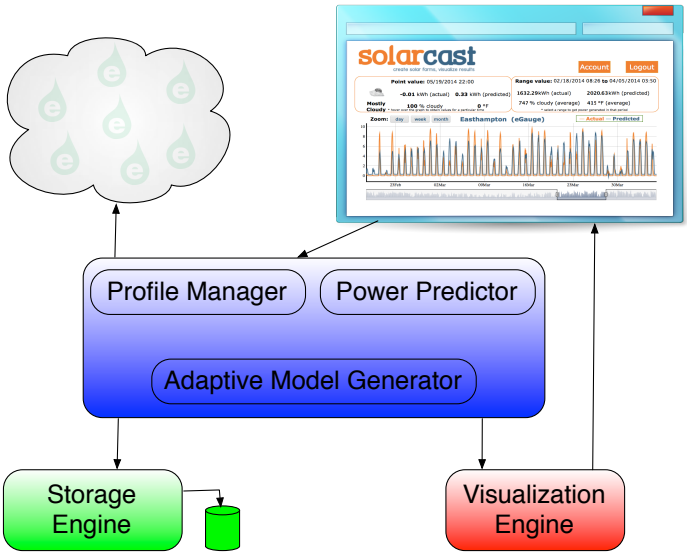


Figure 4.5. SolarCast Web Service Architecture

SolarCast does not require any panel or configuration parameters from the user. Instead, its black-box service automatically learns these parameters, as discussed in the previous section.

SolarCast consists of five primary components, which we detail below: (a) Profile Manager, (b) Visualization Engine, (c) Predictive Model Generator, (d) Power Predictor, and (e) Storage Engine. First and foremost, a user needs to create an account with SolarCast, and then create an installation profile. The installation profile contains the site location and any other optional information provided by the user. The installation profile is the key information for all other components; they operate on a per-profile basis. A user can then have multiple installation profiles, where a profile may be associated with multiple users.

- **Profile Manager**. The profile manager is responsible for managing users and associated profile information. When a user logs in, the profile manager gets the associated profile(s) and other information from the storage engine and calls the visualization engine to display the required information on the user's browser.

- **Visualization Engine**. The visualization engine interacts with the profile manager and power predictor to get the necessary information, such as point forecast, average forecast, and predicted power generation, to render and display on the user's browser. The graphical and intuitive display enables the user to easily grasp the historical, as well as predicted power generation, for any time interval in the future or past.

- **Storage Engine**. The storage engine is responsible for formatting and storing raw historical, as well as forecast, data into a relational database. Further, it also stores customized site-specific forecast models in the database. All other components contact the storage engine for retrieving information, such as historical/forecast data and forecast models. When a user uploads historical power data it also pulls corresponding forecast data from Forecast.io to store in the database.

- **Model Generator.** The adaptive model generator and power predictor are the core components of SolarCast. Whenever a user uploads historical power data for an installation profile, the profile manager first calls the storage engine to store the data and then triggers the adaptive model generator. The model generator gets the stored data for that profile from the storage engine and runs the ML-based adaptive algorithm to generate a custom prediction model for that installation profile. Moreover, the model generator automatically refines the prediction model if the user uploads any new information.

- **Power Predictor**. The power predictor is called when a user sends a request to generate a prediction report for a selected time interval. The power predictor gets the forecasting model from the storage engine, pulls real-time forecast data from Forecast.io, and predicts power generation for the selected interval. The web service calls the visualization engine to format the results and display them to the user; it provides point-by-point predictions as well as average prediction of the weather condition and power output from the installation.

4.3.2 Implementation

We use many open source libraries to build SolarCast and its black box prediction model. We use Django [36], an open source web application framework written in Python, to build SolarCast's web service. The visualization engine uses dygraph [37], a Javascript charting library specifically designed to display time series data, to display solar power predictions to users. We use *scikit-learn* to design our black box prediction model, which is an open source machine learning library for Python. The deep neural network architecture was implemented using Keras [28], an open source deep learning library in python. In addition, we use libraries – SciPy, NumPy, Pandas – from the SciPy stack [7] for data processing. To store users' profiles, prediction models, and dataset we use SQLite, a lightweight disk-based relational database.

Since sensors used by many households report power readings in their local time zone, accounting for daylight savings time in the prediction model is challenging. For this purpose, we convert local time readings to standard Unix time using the Python pytz library [85], which automatically handles the daylight saving issues. To get weather forecasts for any location we use the Forecast API from Forecast.io [43]. Forecast.io provides simple RESTful APIs to retrieve both historical as well as future forecasts of several weather parameters, such as cloud cover, temperature, humidity, precipitation potential, dew point, wind speed, and wind direction, etc. It returns data in the JSON format. Furthermore, we use the National Renewable Energy Laboratory recommended Masters' Algorithm to get the Sun's altitude and azimuth, and the cloudless irradiance at a particular time for a given location. We use the PySolar library [84] that implements the Masters' algorithm.

4.4 Evaluation

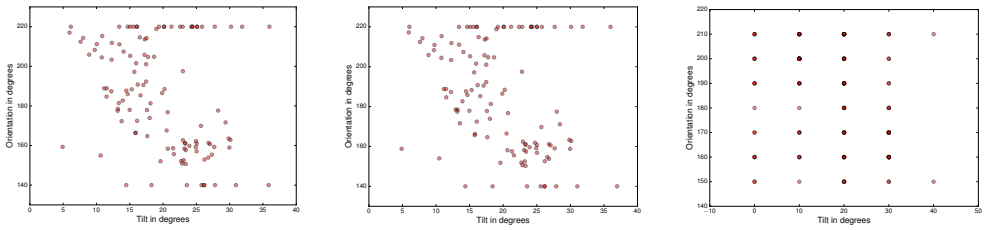
We evaluate our black box prediction model on three geographically diverse datasets. Table 5.2 describes them in terms of their number of installations, duration, data granularity and installed capacity. All the installations in the Pecan Street dataset [4] are located in

Name	*Installations*	*Duration*	*Granularity*	*Size*
Pecan Street	116	2 years	Hourly	5-20kW
Utility	3	2 years	Hourly	.8 to 3.5 MWlevel
$3^{rd} Party$	116	1 year	Hourly	5-150 kW

Table 4.2. Details of the different datasets used in the evaluation

Austin, Texas. The 116 sites from a third-party site are spread across 16 different states in the U.S., whereas the three medium-sized installations are managed by a utility located in the Northeast U.S. Each dataset contains the location – latitude and longitude – and historical power generation readings collected for 12 to 24 months using energy meters (the accuracy is discussed in prior work [19]). In each case, we use the first half of the dataset for training and the next half for testing. For the three sites from the utility dataset and one from the third-party dataset, we had access to real-time data.

We first learn the configuration parameters – i.e. the tilt and the orientation – for each site using an optimization for the constrained least squares curve fit technique and grid search for the deep neural networks. Next, we use the site-specific configuration parameters in our black box models for each site. For each site, the static model built using constrained least squares curve fit, leverages features developed using prevalent weather forecasts, whereas the adaptive approach (again built using constrained least-squares curve-fit) additionally uses immediate past generation data to get a more refined model to predict the next day power generation. The neural network based approach uses features similar to the static approach for deriving a deep neural architecture to learn the complex relationship between the weather parameters and power generation. To compare with an existing machine learning based forecasting technique, we use the SVM-ML model discussed earlier which uses a support vector machines (SVM) [94]. We experimented with 3 different kernels - 1) Linear, 2) Polynomial, and 3) RBF. Of these three kernels, linear kernel performed the best for our problem with the datasets we had. Thus, we have only included the results



(a) Static (Curve Fit) (b) Adaptive (Curve Fit) (c) Grid Search (NN)

Figure 4.6. Tilt and orientation for different solar installation in the Pecan Street dataset.

with the linear kernel. As opposed to predicting irradiance, we directly predict power based on weather forecasts for each day.

4.4.1 Learning Configuration Parameters

As discussed earlier, our method relies on learning configuration parameters, such as the tilt and orientation of the different sites, to build a solar power prediction model. These parameters are learned directly in the optimization function used in the constrained least-squares curve-fit, and using the grid search routine for the deep neural network. These parameters are constrained to be between 0° to 60° and 140° to 220° for tilt and orientation respectively in case of the curve fit method. For the deep neural network, we employ grid search by varying the tilt between 0° and 40° and the orientation between 150° and 210°, both with a step of 10°, to find the values that minimize the average MAPE over the training data.

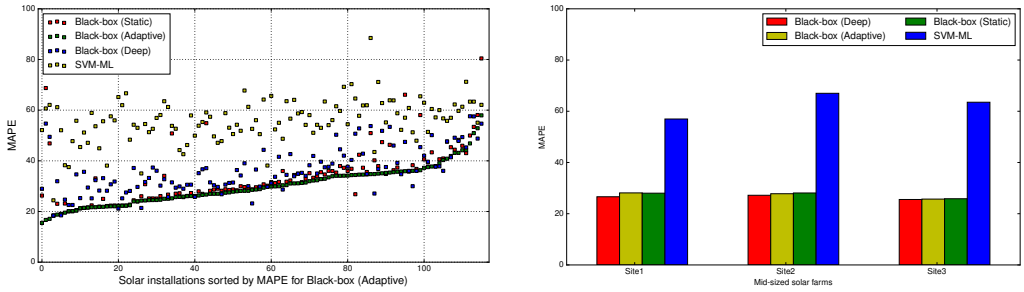
Figure 4.6 shows the tilt and orientation for the three different algorithms for each site in the Pecan Street dataset, where the x-axis is the tilt and the y-axis is the orientation. As the figures show, the tilt and orientation vary greatly across different sites, which highlights the importance of an automatic technique like our black box model to learn the configuration parameters rather than assuming fixed values for all sites.

4.4.2 Model Comparison

- **Hourly predictions:** In this section, we compare the prediction accuracy of three models—our black box static model, our black box adaptive model, our black box neural network model—with the SVM-based prediction model from prior work [94]. We use MAPE to measure the prediction accuracy of each model. Figure 4.7(a) plots MAPE for all four models for 116 rooftop installations for the third-party dataset, while Figure 4.7(b) plots the same for the three open-space medium-sized utility solar farms, each with over .8MW capacity. In Figure 4.7(a), we see that the adaptive approach performs best as it adapts to the dynamic parameters, such as dust, leaves, or pollen, and is slightly better ($\sim$2-3%) than the static model, which only learns the static configuration parameters and is oblivious to the dynamic factors. The Neural Network based approach outperforms the adaptive approach for a few sites by ($\sim$3-5%) but overall seems to produce inferior results. In Figure 4.7(b), we find that the neural network approach performs the best for all three locations by ($\sim$.5-2%). For both graphs, the SVM-ML model performs worse because it constructs a separate model for each time of the day, thereby using just part of the training data. Further, it does not capture the yearly variation in the position of the sun for the same time of the day. For example, at noon, the Sun is closer to zenith during summer than during winter for a given location. By normalizing the features using the cloudless irradiance, we address both these shortcomings.

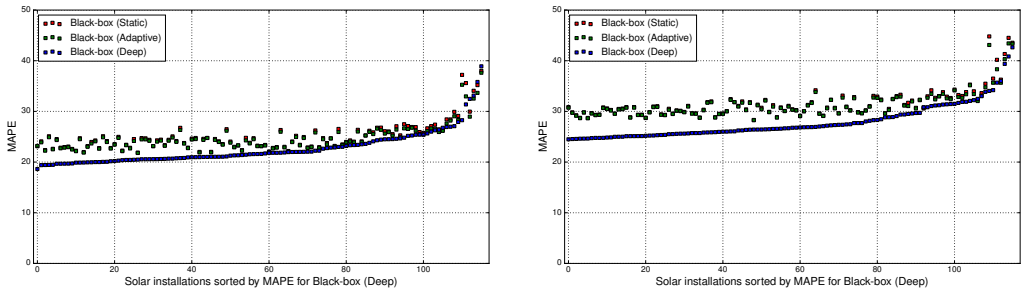
  Figure 4.8(a) shows the comparison between our three black box approaches on the Pecan Street dataset with hour-level forecasts. For almost half the sites the deep neural network based approach outperforms the static and the adaptive approach by ($\sim$3-5%). In most other cases, the performance is ($\sim$1-3%) better.

- **Minute-level predictions:** The experiments above were based on hour-level predictions of solar generation based on forecasts released by the National Weather Service each hour. However, our techniques are agnostic to the data resolution: as long as



(a) Small rooftop installations (b) Medium sized installations

Figure 4.7. Prediction error for various prediction models over 6 months for 3^{rd} Party and Utility dataset



(a) Hour level forecasts (b) Minute level forecasts

Figure 4.8. Prediction error for various prediction models for each site in Pecan Street dataset over 1 years.

we have a forecast available at a particular resolution, we can apply our techniques to predict future solar output at that resolution. Forecast.io provides basic minute-level forecasts one hour into the future for each specific location in the U.S., e.g., as a specified longitude and latitude. These forecasts only predict rain (and its intensity) and do not include the multiple metrics in a typical National Weather Service weather forecast. As a result, minute-level predictions may not capture reduced solar generation due to clouds that do not produce rain.

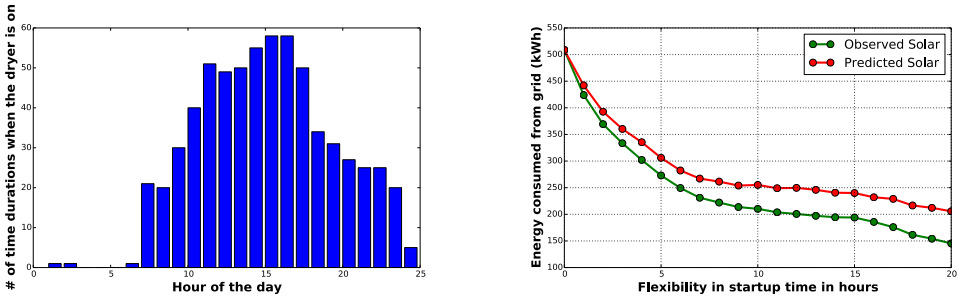
However, we apply our techniques to them to demonstrate the flexibility of apply predictions at high data resolutions. Such predictions might be useful for utility operators, which have to balance grid supply and demand in real time, e.g., second-to-second and minute-to-minute. Figure 4.8(b) shows the comparison between our three black box approaches on the Pecan Street dataset with minute level forecasts. The error is slightly higher for all sites compared to the hour level forecasts due to the inclusion of only a single forecast parameter (rain intensity). However, we again find that the deep neural network-based approach performs better than the static and the adaptive black box approaches. In most cases, the difference in performance is ($\sim$5-7%).

4.5 Case Study

In this section, we explore how households can leverage our black box prediction model in two case studies: scheduling elastic background loads to reduce electricity bills, and providing accurate predictions of charging profiles to customers at a solar-powered EV charging station.

4.5.1 SolarCast in Smart Homes

To maximize green energy penetration homeowners can schedule certain elastic loads, such as plug-in EVs, washing machines, or clothes dryers, to run when solar energy is abundant. We experiment with sunny scheduling of a dryer in a smart home located in the state of Massachusetts. The home's power usage varies from 0 to 18.88 kW with an average of 1.38 kW. The solar power generation varies from 0 to 9.71 kW with an average of 1.43 kW. Thus, the house is a net generator of electricity. We have per-hour data with average power for the solar generation, total electricity usage (excluding the dryer) and an additional load of a dryer. The dryer is running for 652 hourly intervals out of the overall 8258 hours (49 weeks). Note that a single load can run for multiple hourly timeslots.



(a) (b)

Figure 4.9. Frequency distribution for hour of day when the dryer is run (a) and grid demand for start time flexibility (b).

Figure 4.9(a) shows the frequency distribution for the hour of the day when the dryer runs. The figure demonstrates that the dryer is typically operational in the afternoon.

For this case-study, we make an assumption that the magnitude of all loads, including that of the dryer, is known beforehand. We employ an online scheduling algorithm that allocates loads to the earliest contiguous timeslots where the minimum load from the grid is drawn based on solar predictions that have been learned online using day-ahead forecasts. In this algorithm, we bring flexibility in scheduling by running the load in a timeslot of $\pm k$ hours to the actual time. While allocating dryer loads, we ensure multiple loads are not scheduled during the same timeslot. We compute excess power for timeslot j by subtracting the grid electricity from the predicted solar power.

The overall energy consumed by the dryer is 863.54 kWh. With the existing schedule, the total power drawn from the grid is 508.63 kWh. In Figure 4.9(b), we show the results of running the algorithm with observed and predicted solar power generation values with varying flexibility. Even though most of the dryer loads are scheduled during afternoons when solar intensity is strong, there is a substantial reduction in electricity drawn from the grid by having a flexibility of few hours.

In summary, our results show that smart homes can leverage SolarCast's predictions to better schedule elastic loads to align with solar generation. In this case, our smart home

reduces its grid energy demand by 40% by providing flexibility of $\sim$5 hours to a dryer's startup time.

4.5.2 SolarCast in Smart EV charging

Over the past few years, EVs have gained popularity because of their appeal as an environmentally-friendly mode of transportation. EVs have a tremendous potential to reduce our carbon footprint and our dependence on fossil fuels. However, as discussed in prior work [105], these EVs can be more detrimental to the environment as they require charging batteries with low efficiency from the grid, which primary consists of carbon-based power plants. To offset the environmental impact of these cars, we must ensure that they are charged with green energy sources, such as solar. However, as discussed earlier in the chapter, solar energy is intermittent and at many times unreliable due weather changes.

In this case study, we explore the possibility of a solar-powered EV charging station equipped with an array of panels that can help customers by providing an estimate on the amount of energy that can be provided using SolarCast's power predictions. This station could be a parking lot in a company where employees can park their vehicles for the day. Here, we simulate a charging station by using multiple solar rooftop installations at houses from the Pecan Street dataset [79]. We selected five different EVs containing two Teslas, two Chevrolet Volts and one Nissan Leaf from the Pecan Street dataset, which uses the solar power from our simulated charging station. Initially, the user provides the charging required for their EVs at the start of the day. Based on SolarCast's power predictions, we provide an estimate on the amount of charging for that day.

Our aim is to provide best-effort charging, where we provide equal access to the available solar power to all parked EVs, while maintaining following constraints - i) the car batteries have a certain limit and cannot be overcharged, and ii) the EVs cannot draw more energy than the amount generated by the solar installation. Further, as we do not have access to battery charging levels, we assume that the maximum charging allowed is equal

EV Name	*Overall Yearly Demand (kWh)*	*Max Charging rate (kW)*
Tesla1	1651.31	6.68
Tesla2	2185.87	6.83
Volt1	1260.26	3.37
Volt2	1469.98	3.39
Leaf1	1218.02	3.77

Table 4.3. Details of the different EVs used in the case-study

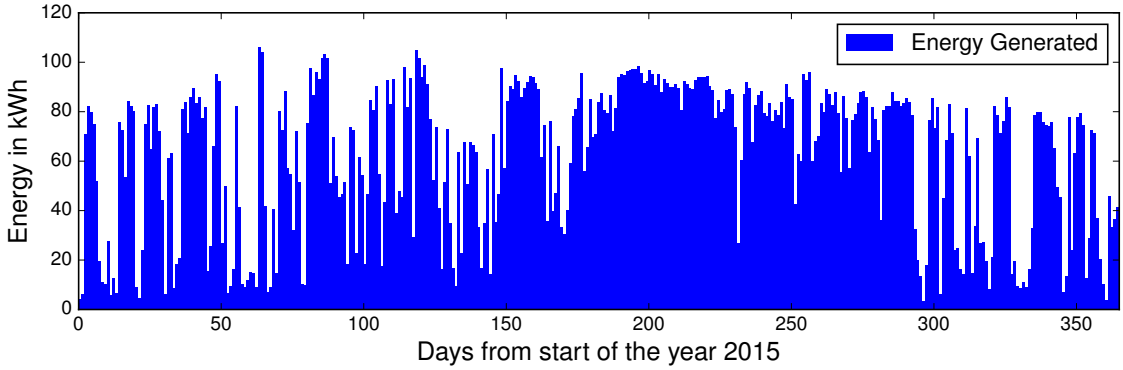


Figure 4.10. Solar energy produced by the smart charging station over the year 2015

to the amount of demand shown in the data (see Figure 4.11). We assume that we have EVs over the entire duration of the day when the sun is above the horizon. We ran the experiment for charging these EVs over the period of a whole year from 1^{st} January to 31^{st} December 2015 using minute level forecasts for the smart charging station (rooftop installation). Figure 4.10 shows the energy produced by the solar charging station over the period of a year. Table 4.5.2 describes the maximum charging rate and the energy consumed over the period of a year by the EVs. Figure 4.11 shows the demand profile of the different EVs over the same period.

We observed that there was a demand of 7785.46 kWh from the five EVs. Through our best effort charging, we were able to satisfy 5423.38 kWh of the EV demand. Further, Figure 4.12 shows the difference in energy between the promised charging provided by SolarCast estimates and the delivered charging using the available solar power for the whole

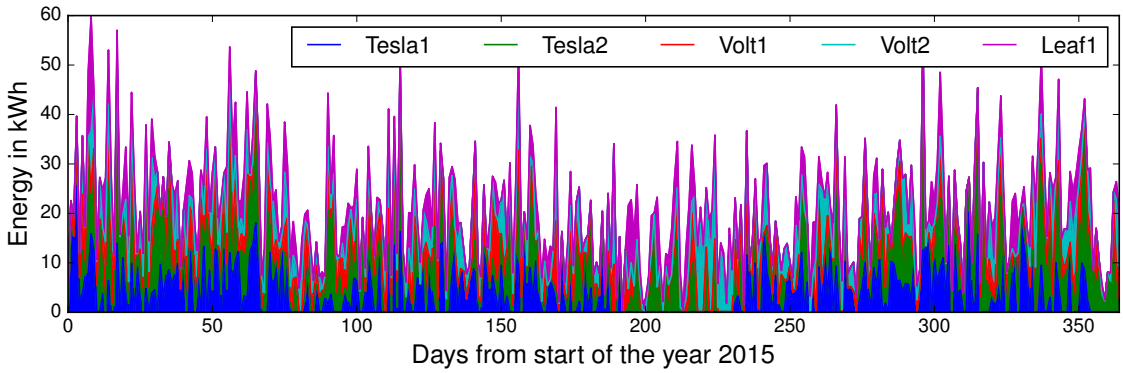


Figure 4.11. Energy demand profile of the different EVs in our study over the period of the year 2015

year for each of the 5 EVs. For the two Teslas, we were able to completely satisfy the demand for approximately 240 days. For the other EVs, we were able to satisfy the energy demand for around 300 to 330 days in a year. The solar charging station had an average of 1.21 kWh of the absolute difference between the promised and the delivered energy over the average daily charge of 14.86 kWh per day.

In summary, our results show the following -

- Of the total demand of 7785.46 kWh from the five EVs, our simulated solar charging station was able to satisfy 5423.38 kWh of the EV demand using best effort charging. Further, the EVs were completely charged for approximately 230 to 330 days in a year.

- The mismatch (absolute difference) between promised and delivered energy for the solar charging station was on average 1.21 kWh per day over the average daily charge of 14.86 kWh per day.

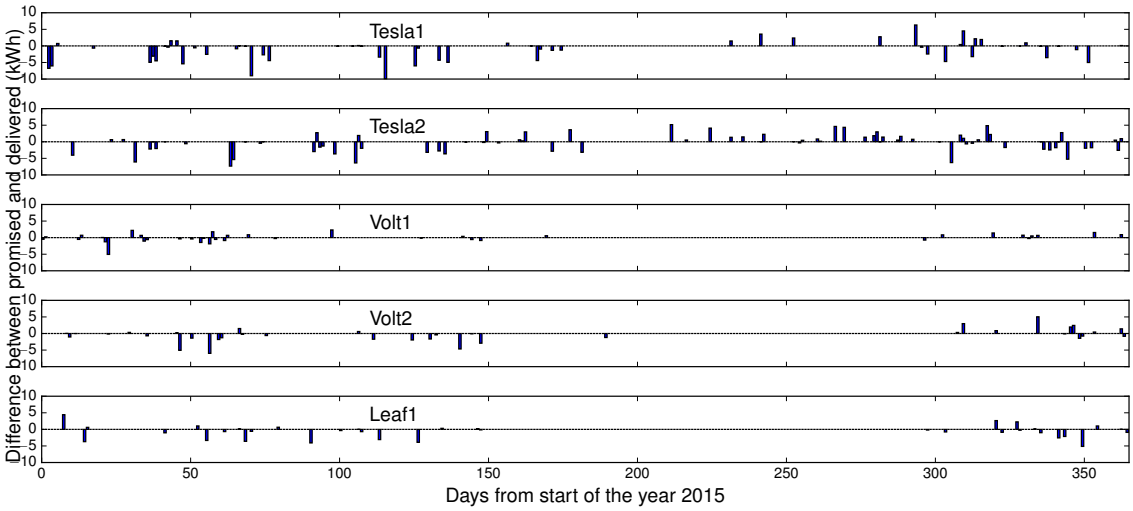


Figure 4.12. Mismatch between the promised and the delivered energy for the different EVs over the whole year

4.6 Limitations of the model

In this section, we discuss the limitations of our models due to inaccuracies in weather forecasts. As our models are a function of weather forecasts, any inaccuracy in them will result in errors in solar power prediction.

Cloud cover is a major contributor to the intermittency of solar power. To present the impact of inaccuracies in cloud cover, we looked at two set of days - i) clear sky days with no cloud cover (according to weather forecasts), and ii) overcast days having similar cloud cover levels. While constructing these sets, we ensured that the other weather parameters, such as precipitation, visibility, and temperature, are also not very different for days within the set. Further, all the days in a given set were chosen such that they are within 25 days of each other. Figure 4.13 shows the recorded solar power and our predictions for the four clear sky days and the three overcast days. Furthermore, to evaluate our predictions, we also plot ground truth irradiance with average cloud cover for that day. The data shown in the figure is from a rooftop installation in the Pecan Street dataset. The irradiance data for Austin (location of the site) is collected from a wunderground [102] weather station.

4.6.1 Clear Sky days

Figure 4.13(a) shows the four clear sky days. Apart from the second day, our algorithm has MAPE between ∼6-10%. However, on the second day, as corroborated by the irradiance data, the second half of the day has reduced irradiance, which was missed by our weather forecasts. In this case, our MAPE increased to 22.5%.

4.6.2 Overcast days

Figure 4.13(a) shows the three cloudy days. According to our forecast data, all the three cloudy days were had similar cloud cover throughout the day with increased cloud cover in the middle of the day. For the first two days, we observe that increased cloud cover in the late morning causes a dip in solar production which was partially captured by our model resulting in a MAPE of ∼25-31%. However, on the third day our model over-predicts, resulting in a large MAPE value. The ground truth irradiance shows that the day was more overcast than suggested by weather forecasts.

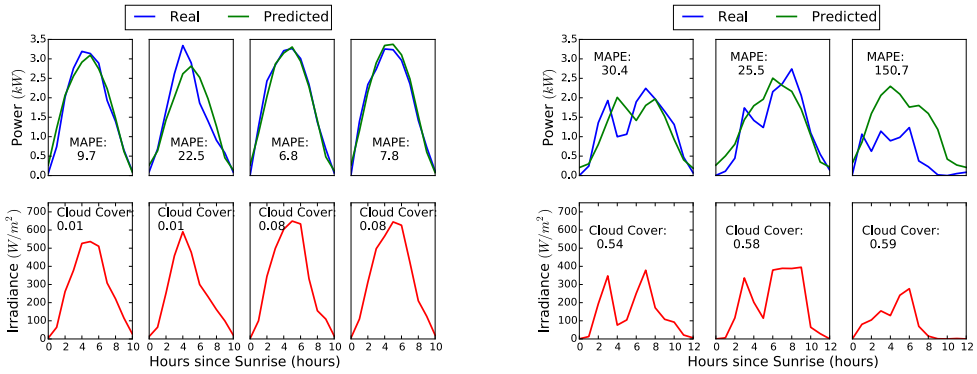
Thus, our key observations are as follows:

1. As our models are a function of weather forecasts, any error in them would translate to prediction error in solar power.

2. Even with correct cloud cover information, the MAPE values associated with overcast days was higher than those for the clear sky days. This clearly suggests that cloud cover is not a reliably accurate predictor of solar power (e.g., 50% cloud cover could mean light uniform clouds or scattered clouds covering half the sky).

4.7 Related Work

Prior work on solar forecasting focuses on predicting solar power for a particular solar panel installation or only a few installations with well-known characteristics. They either predict solar irradiance and get power from the irradiance or directly predict the power.



(a) Sunny days

(b) Cloudy days

Figure 4.13. Error analysis for the two sets of days

In both cases, they assume all panel and configuration parameters are known in advance. However, these are often unknown for a typical rooftop installation. Lorenz et al. [70] and Huang et al. [52] provide a comprehensive comparison of different solar irradiance and power prediction techniques, respectively. These techniques can be classified as persistence method, satellite data/imagery method, numeric weather prediction (NWP) method, statistical method and hybrid method. Each of these methods is suitable for different time horizons. For example, the persistence model is ideal for short-term forecasting (1 hour ahead), whereas statistical methods are more effective for medium-term forecasting (1 to 36 hours ahead). Yona et al. [104] use a neural network model to forecast solar irradiance; they then use a site-specific irradiance $\rightarrow$ power model to forecast power generation.

Tao et al. [98] propose a nonlinear autoregressive exogenous model that uses installation parameters, such as tilt and orientation, to forecast day-ahead power generation. The input layer of the model includes cloudless irradiance for the next day from 6am to 6pm. To predict power at any arbitrary time it further requires additional input nodes with adequate training. These restrictions exist for [29] that uses Elman Neural Networks with an input layer very similar to that of related work [98]. Mandal et al. [73] presents a hybrid model that uses wavelets and Neural Networks. Moreover, all of these techniques have used a

single site and limited dataset ($\sim$4 days) for evaluation. Apart from neural network models, machine learning (ML) based statistical techniques [49, 69, 94] are also gaining popularity in the past decade. These techniques typically use weather forecasts and historical data, to predict power generation at short time scales.

Unlike prior work, SolarCast does not require panel and configuration parameters; it automatically learns these parameters from minimal historical data. Further, its black box architecture allows it to scale across a number of sites, ranging from rooftop installations to large solar farms. To our knowledge, we are the first to evaluate our black box model on datasets with many solar sites with different characteristics.

4.8 Conclusion

In this chapter, we present a black box approach for forecasting solar power generation. Our black box model only needs the location and minimal historical data from any solar panel installation to design a custom site-specific prediction model. We evaluate this approach using two different machine learning approaches—one based on a least-squares curve-fit and one based on a deep neural network—-across multiple datasets that include more than one hundred solar deployments each (spread across multiple geographic locations). Importantly, our approach learns much faster than prior approaches by normalizing each data point at each point in time relative to the weather, e.g., 50% cloud cover at 12pm and 6pm has the same affect on the percentage of solar output. This normalization enables our approach to applying all the data to a single model. In addition, unlike prior techniques, our adaptive black box model also accounts for the dynamic factors, such as snow, dust, and pollen, which is evident from its low prediction error compared to prior machine learning based prediction model. Finally, we present two application case studies. Our first application case study shows how a smart home can exploit SolarCast's services to schedule elastic loads and reduce electricity bills. As an example, we show that by simply providing a little flexibility for a dryer's start time, the homeowner can reduce grid energy demand by up to

40%. We then evaluate a smart solar-powered charging station, which can optimally charge the maximum number of electric vehicles (EVs) on a given day, and show that SolarCast can provide EV owners the amount of energy they can expect to receive from solar energy sources.

CHAPTER 5

ANOMALY DETECTION IN SOLAR POWER GENERATIONS

This chapter proposes SolarClique, a data-driven approach to flag anomalies in solar power generation. SolarClique utilizes a graphical model formulation to distinguish between anomalies affecting solar output and weather-related factors such as cloud cover. Moreover, this approach neither relies on expensive instrumentation nor does it require external inputs such as weather data. In this chapter, we also discuss the approach in detail and show how it exploits correlations in solar power generation from geographically nearby sites to predict the expected output of a site and flag anomalies. Further, this chapter presents an extensive evaluation of this approach. Specifically, we show that it can scale to sparsely populated regions, where there are few solar installations.

5.1 Motivation

Technological advances and economies of scale have significantly reduced the costs and made solar energy a viable renewable alternative. From 2010 to 2017, the average system costs of solar have dropped from $7.24 per watt to $2.8 per watt, a reduction of approximately 61% [44]. At the same time, the average energy cost of producing solar is 12.2¢ per kilo-watt and is approaching the retail electricity price of 12¢ per kilo-watt [3]. The declining costs have spurred the adoption of solar among both utilities and residential owners.

Recent studies have shown that the total capacity of small-scale residential solar installations in the US reached 7.2 GW in 2016 [5]. Unlike large solar farms, residential installations are not monitored by professional operators on an ongoing basis. Consequently,

anomalies or faults that reduce the power output of residential solar arrays may go undetected for long periods of time, significantly reducing their economic benefits. Further, large solar farms have extensive sensor instrumentation to monitor the array continuously, which enables faults or anomalous output to be determined. In contrast, residential installations have little or no sensor instrumentation beyond displaying the total power of the array, making sensor-based monitoring and anomaly detection infeasible in such contexts. Adding such instrumentation increases the installation costs and is not economically feasible in most cases.

However, the primary challenge in designing such an application is to handle intrinsic variability of solar and site-specific idiosyncrasies. Several factors affect the output of a solar panel — such as weather conditions, dust, snow cover, and shade from nearby trees or structures, temperature, etc. We refer to such factors as transient factors since they temporarily reduce the output of the solar array. For instance, a passing cloud may briefly decrease the power output of the panel but doesn't reduce the solar output permanently. Similarly, shade from nearby buildings or trees can be considered transient factors as they reduce the output temporarily and may occur only at certain periods of the day.

Interestingly, some transient factors, such as overcast conditions, impact the output of all arrays in a geographical neighborhood, while other factors such as shade from a nearby tree impact the output of only a portion of the array. In addition, factors such as malfunctioning solar modules or electrical faults also reduce the output of a solar array, and we refer to them as *anomalies* — since human intervention is needed to correct the problem. Prior studies have shown that such factors can significantly reduce the power output by as much as 40% [10, 34, 45]. In our work, we need to distinguish between the output fluctuations from transient and anomalous factors. Further, site specific idiosyncrasies (such as shade, tilt/orientation of panels) need to be considered when exploiting the correlation between solar arrays in a region.

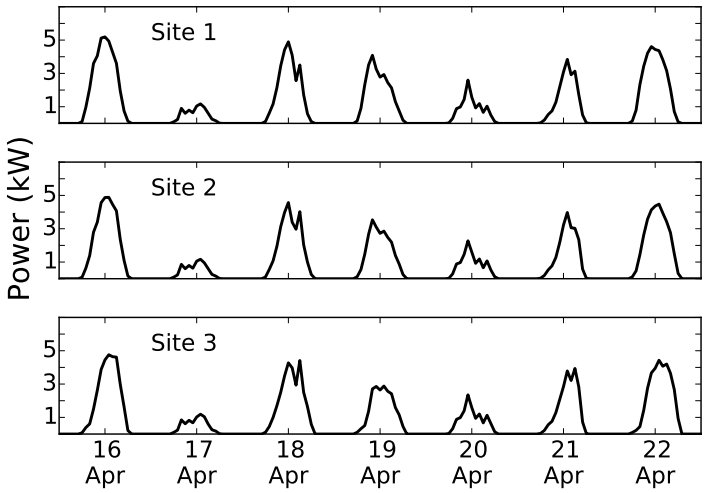


Figure 5.1. Power generation from three geographically nearby solar sites. As shown, the power output is intermittent and correlated for solar arrays within a geographical neighborhood.

Naive approaches such as labeling a solar installation as anomalous whenever its power output remains "low" for an extended period do not work well. Since drops in power output may be caused due to cloudy conditions, depending on the weather, the solar output may remain low for days. Labeling such instances as anomalies may result in many false positives. Since the challenge lies in differentiating drops in power output due to transient factors (i.e., factors that impact power output temporarily) and those that are anomalies (i.e., factors that may require human intervention), we need a new approach for detecting solar anomalies using geographically nearby sites.

The rest of the section is organized as follows. We present a graphical model representation for our setup that models the confounding variables. Next, we discuss how our algorithm removes the confounding factors and detects anomalies in solar generation.

5.2 Graphical model representation

We first introduce the intuition behind our approach to detect anomalous power generation in a solar installation. Our primary insight is that other geographically nearby sites can

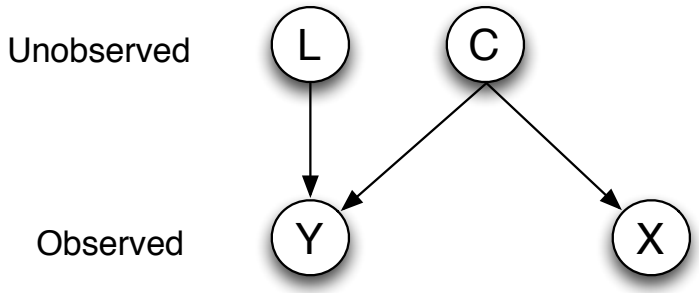


Figure 5.2. Graphical model representation of our setup.

predict the solar output potential, which can then reveal issues in a given site. Since factors such as the amount of solar irradiance (e.g., due to cloudy conditions) are similar within a region, the power output of solar arrays in a geographical neighborhood is usually correlated. This can be seen in the power output from three different solar installation sites in the same geographical neighborhood (see Figure 5.1). As seen, the solar arrays tend to follow a similar power generation pattern. So we can use the output of a group of sites to predict the output of a specific site and flag anomalies if the prediction significantly deviates.

We hypothesize that predicting the output using geographically nearby sites can "remove" the effects of confounding factors (i.e., common factors). By accounting for confounding factors, the remaining influence on power generation can be attributed to local factors in the solar installation. The local factors may include both transient local factors and anomalies. Thus, any irregularity in power generation, after accounting for confounding and transient local factors, must be due to anomalies in the installation. For example, cloudy or overcast conditions in a given location have a similar impact on all solar panels and will reduce the power output of all sites. However, a malfunctioning solar module in a site (a local event) will observe a higher drop in power output than others. If the drop in power due to cloudy conditions (a confounding factor) along with transient local factors is accounted for, any further drop in power can be attributed to anomalies. Our approach follows this intuition to detect anomalies in a solar installation.

Our work is inspired by a study in astronomy, wherein *Half-Sibling Regression* technique was used to remove the effects of confounding variables (i.e., noises from measuring instruments) from observations to find exoplanets [90]. We follow a similar approach to model and detect anomalies in a solar installation.

Let C, L, X and Y be the random variables (RVs) in our problem. Here, Y refers to the power generated by a candidate solar installation site. X represents the power produced by each of the geographically nearby solar installations (represented in a vector format). While C represents the confounding variables that affect both X and Y, the variable L represents site-specific local factors affecting a candidate site. These local factors include both transient factors and anomalies that affect a candidate site. In our setup, both X and Y are observed variables (as power generation of a site can be easily measured), whereas C and L are latent unobserved variables. Figure 5.2 depicts a directed graphical model (DAG) that illustrates the relationship between these observed and unobserved random variables.

We are interested in the random variable L which represents anomalies at a given site. As seen in the figure, since both L and C affect the observed variable Y, without the knowledge of C it is difficult to calculate the influence of L on Y. Clearly, X is independent of L as variable L impacts only Y. However, we note that C impacts X and when conditioned on Y, Y becomes a *collider*, and the variables X and L become dependent [78]. This implies that X contains information about L and we can recover L from X.

To reconstruct the quantity L, we impose certain assumptions on the type of relationship between Y and C. Specifically, we assume that Y can be represented as an additive model denoted as follows:

$$Y = L + f(C) \tag{5.1}$$

where f is a nonlinear function and its input C is unobserved. Since L and X are independent, variable X cannot account for the influence of L on Y. However, X can be used to approximate $f(C)$, as C also affects X. If X exactly approximates $f(C)$, then $f(C) = E[f(C)|X]$, and we can show that L can be recovered completely using (5.1).

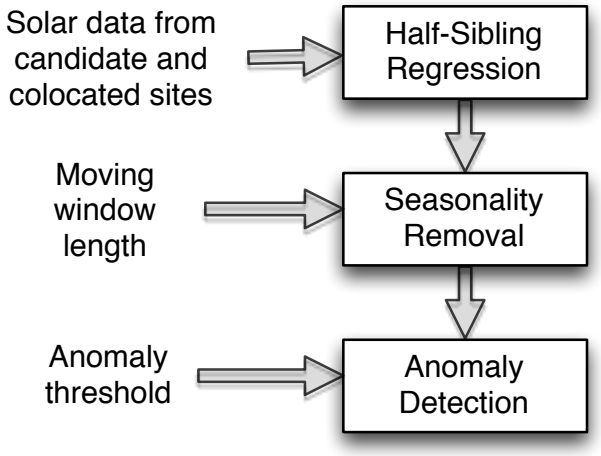


Figure 5.3. An overview of the key steps in the SolarClique algorithm.

Even if X does not exactly approximate $f(C)$, in our case, X is sufficiently large to provide a good approximation of $E[f(C)|X]$ up to an offset. A more detailed description of the approach is given in [90]. Thus, using X to predict Y (i.e., $E[Y|X]$), $f(C)$ can be approximated and removed from (5.1) to estimate $\hat{L}$ as follows:

$$\hat{L} \coloneqq Y - E[Y|X] \tag{5.2}$$

where $\hat{L}$ is an estimate of the local factors that may include both transient local factors and anomalies.

5.3 SolarClique Algorithm

We now present our anomaly detection algorithm called *SolarClique*. Figure 5.3 depicts an overview of the different steps involved in the *SolarClique* algorithm. First, we use the *Half-Sibling Regression* approach to build a regression model that predicts the solar generation of a candidate site using power output from geographically nearby sites. Next, we remove any seasonal component from the above regression model using time series de-

composition. Finally, we detect anomalies by analyzing the deviation in the power output. Below, we describe these three steps in detail.

5.3.1 Step 1: Remove confounding effects

The first step is to build a regression model that predicts the power generation output Y of a candidate site using X, a vector of power generation values from geographically nearby solar installations. As mentioned earlier, the regression model estimates $E[Y|X]$ component in the additive model shown in (5.2). Since Y is observed, subtracting the $E[Y|X]$ component determines the L component.

Standard regression techniques can be used to build this regression model. The regression technique learns an estimator that best fits the training data. Instead of constructing a single regression model, we use *bootstrapping* — a technique that uses subsamples with replacement of the training data — which gives multiple regression models and the properties of the estimator (such as standard deviation). We use an ensemble method, wherein the mean of the regression models is taken to estimate the $E[Y|X]$ in the testing data. Finally, we remove the confounding component $E[Y|X]$ from Y to obtain an estimate of $\hat{L}_t \; \forall t \in T$ in the testing data. The final output of this step is an estimate $\hat{L}_t$ and the standard deviation (σ_t) of the estimators.

5.3.2 Step 2: Remove seasonal component

As discussed earlier, the solar output of a site is affected by both common (i.e., confounding) and local factors. Using the *Half-Sibling Regression* approach, we can account for the *transient* confounding factors such as weather changes. However, we also need to account for *transient* local factors, such as shade from nearby trees, which may temporarily reduce the power output at a specific time of the day. Since variable $\hat{L}_t$ include both transient local factors and anomalies, we need to remove the local factors to determine the anomaly $\hat{A}_t$.

We note that the time period of such occlusions (those from nearby trees or structures) may not vary much on a daily basis. This is because the maximum elevation of the sun in the sky varies by less than 2° over a period of a week[1] on average. Using time series decomposition techniques over short time intervals (e.g. one week), such seasonal components (i.e the pattern occurring every fixed period) can be removed. Thus, we perform a time series decomposition to account for transient local factors as follows. We compute the seasonal component and remove it from $\hat{L}_t$ only when $\hat{L}_t$ is outside the confidence interval 4σ and on removal of the seasonal component, $\hat{L}_t$ doesn't go outside the confidence interval. After removal of the seasonal component, if any, we obtain $\hat{A}_t$ from $\hat{L}_t$ as our final output.

5.3.3 Step 3: Detect Anomalies

We use the output $\hat{A}_t$ (from Step 2) and the standard deviation σ_t (from Step 1) to detect anomalies in a candidate site. Specifically, we flag the day as anomalous when three conditions hold. First, the deviation of $\hat{A}_t$ should be significant, i.e., greater than four times the standard deviation. Second, the anomaly should occur for at least k contiguous period. Finally, when the period t is during the daytime period (not including the twilight). Thus, an anomaly can be defined as follows:

$$anomaly = (\hat{A}_t < -4\sigma_t) \wedge ... \wedge (\hat{A}_{t+k} < -4\sigma_t) \qquad \forall t \in T \tag{5.3}$$

where T denotes the time during the daytime period.

Based on our assumption that $\hat{A}_t$ is Gaussian, it follows that the odds of an anomaly are very high when the deviation is more than 4σ. These anomalous values belong to the end-tail of the normal distribution. The second condition (i.e., contiguous anomaly period)

[1]The sun directly faces the Tropic of Cancer (+23.5°) on the summer solstice. Whereas, it faces the Tropic of Capricorn (-23.5°) on the winter solstice. Thus, over half the year (26 weeks) the maximum elevation of the sun changes by ≈47°, i.e., <2° per week.

ensures that the drop in power output is for an extended period. In practice, depending on the data granularity, the contiguous period can range from minutes to hours. Clearly, we would like to detect anomalies during the period when sunlight is abundant. During the night or twilight, the solar irradiation is very low to provide any meaningful power generation. Thus, we choose the daytime period in our algorithm for anomaly detection.

5.4 Implementation

We implemented our SolarClique algorithm in python using the SciPy stack [7]. The SciPy stack consists of efficient data processing and numeric optimization libraries. Further, we use the regression techniques in the scikit-learn library to learn our models [80]. The scikit-learn library comprises various regression tools, which takes a vector of input features and learn the parameters that best describe the relationship between the input and the dependent variable. Additionally, we use Seasonal and Trend decomposition using Loess (STL) technique to remove the seasonality component [30]. The STL technique performs a time series decomposition on the input and deconstructs it into trend, seasonal, and noise components.

5.5 Evaluation Settings

5.5.1 Dataset

For evaluating the efficacy of SolarClique, we use a public dataset available through the Dataport Research Program [4]. The dataset contains solar power generation from over hundred residential solar installations located in the city of Austin, Texas. The power generation from these installations are available at an hourly granularity. Table 6.3 shows the key characteristics of the dataset. For our case study, we selected those homes that have contiguous solar generation data, i.e., no missing values, for an overlapping period of at least two years. Based on this criteria, we had 88 homes for our evaluation in the year 2014 and 2015.

Number of solar installations	88
Solar installation size (kW)	0.5 to 9.3
Residential size (sq. ft.)	1142 to 3959
Granularity	1 hour
Year	2014, 2015

Table 5.1. Key characteristics of the dataset.

5.5.2 Evaluation Methodology

We partitioned our dataset into training and testing period. We used the first three months of data to train the model, and the remaining dataset for testing (21 months). Further, for bootstrapping, we sample our training dataset by randomly selecting 80% of the training samples with replacement. These samples are then used to build the estimator, and we repeated this step 100 times to learn the properties of the estimator. To build our model, we used five popular regression techniques namely Random Forest (RF), k-Nearest Neighbor (kNN), Decision Trees (DT), Support Vector Regression (SVR), and Linear Regression (LR). Finally, we selected the contiguous period as $k = 2$ (see Step 3 of our algorithm) since our data granularity is hourly. Unless stated otherwise, we use all homes in our dataset for our evaluation.

5.5.3 Metrics

Since the installation capacity can be different across solar panels, it may not be meaningful to use a metric such as Root Mean Squared Error (RMSE). This is because the magnitude of the error may be different across predictions. Thus, we use Mean Absolute Percentage Error (MAPE) to measure the regression model's accuracy in predicting a candidate's power generation. MAPE is defined as:

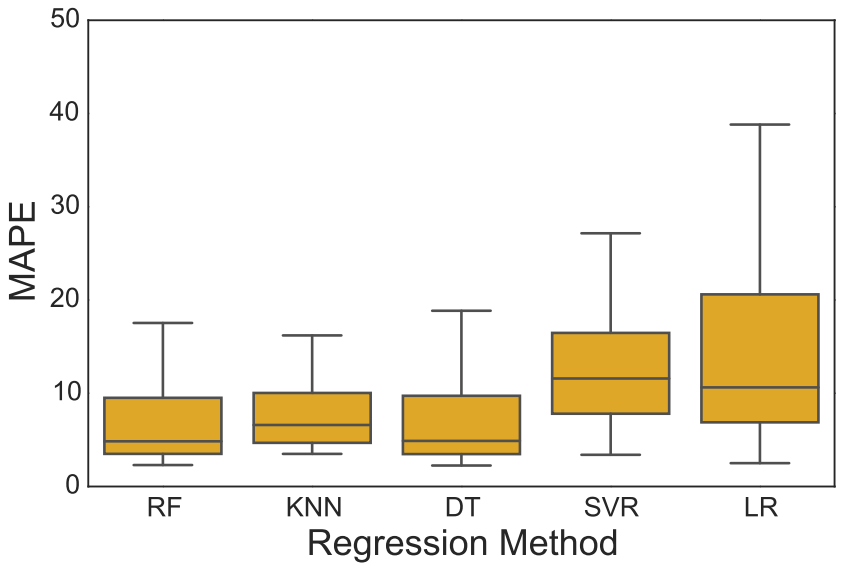


Figure 5.4. Performance of different regression techniques used to predict the power generation of a site.

$$MAPE = \frac{100}{n} \sum_{t=1}^{n} \left| \frac{y_t - p_t}{\bar{y}_t} \right| \tag{5.4}$$

where y_t and p_t are the actual and predicted value at time t respectively. $\bar{y}_t$ represents the average of all the values and n is the number of samples in the test dataset.

5.6 Experimental Results

Below, we summarize the results of using SolarClique on the Dataport dataset.

5.6.1 Prediction performance using geographically nearby sites

We compare the accuracy of the five regression techniques used to predict the power generated at a candidate site (Y) using the data from nearby sites (X). Figure 5.4 shows the spread of the MAPE values for the regression techniques used for all the 88 sites. Random Forest and Decision Trees show the best performance closely followed by k-NN with average MAPE values of approximately 7.81%, 7.87%, and 8.94% respectively. Linear Regression, on the other hand, shows poor accuracy with an average MAPE of 19%.

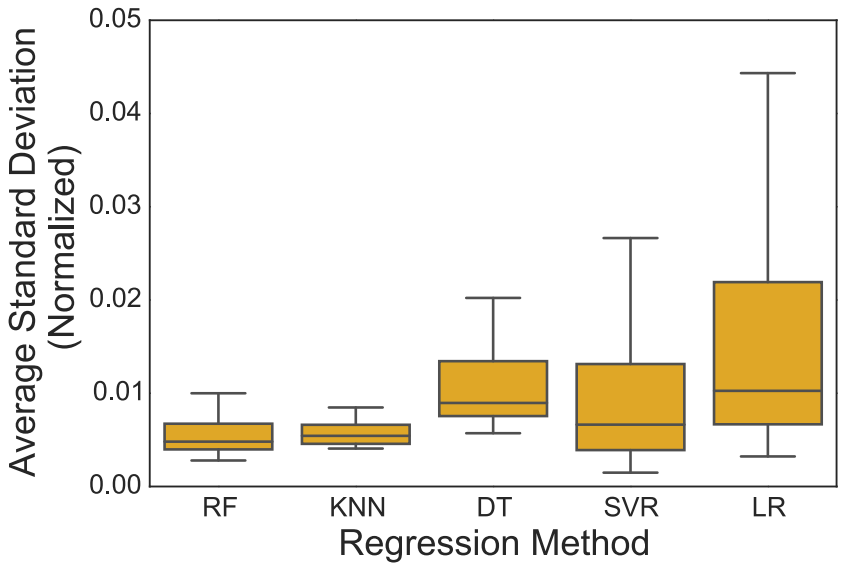


Figure 5.5. Mean standard deviation of predictions for different regression techniques

As discussed earlier, our approach uses bootstrapping to generate the standard deviation values for each prediction. Note that a small standard deviation means tighter confidence interval and indicates that the regression technique has a consistent prediction across runs. Figure 5.5 shows the mean value of standard deviation over all the testing samples normalized by the size of the solar installation. We observe that RF and k-NN have tight confidence intervals, while LR has considerably wider bounds. In particular, we observe that the average standard deviation of RF and k-NN is 0.0032 and 0.0059 using all the sites, respectively. In comparison, the average standard deviation of LR is 0.0078. Since RF performs better than other regression techniques, we use RF for the rest of our evaluation.

5.6.2 Impact due to the number of geographically nearby sites

We now focus on understanding the minimum number of geographically nearby sites to accurately predict the power generated at the candidate site. As discussed earlier, the power output of geographically nearby sites are used as input features to build the regression model. Since in this experiment we are not interested in analyzing the confidence intervals, we use the entire training data to build the model (i.e., no bootstrapping). We

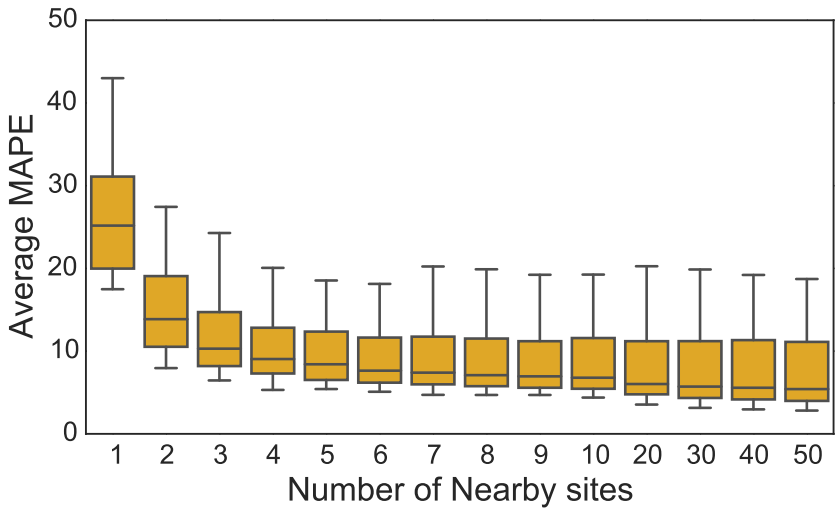


Figure 5.6. Average MAPE diminishes with increase in the number of geographically nearby sites.

vary the number of geographically nearby sites from 1 to 50 and for each value, we build 100 different models learnt from choosing random combinations of nearby sites.

Figure 5.6 shows the spread of average MAPE values as we vary the number of geographically nearby sites used for all 88 sites. We use the Random Forest regression technique to build the model. As expected, the average MAPE value reduces when more number of geographically nearby sites are used to predict the output. Note that as the nearby sites increase, the variations in nearby sites cancel out, which provides a more robust regression model. This suggests that an increase in the nearby site can improve the accuracy of the power generation model of a candidate site. We also note that the reduction in MAPE diminishes as the number of geographically nearby sites increases. With at least five randomly chosen geographically nearby sites, we observe that the MAPE is around 10%. This indicates that our algorithm can be effective in sparsely populated regions such as towns/villages, having few solar installations.

Next, we analyze the variability in performance of the different models as the number of geographically nearby sites increases. Figure 5.7 shows the spread of the standard deviation

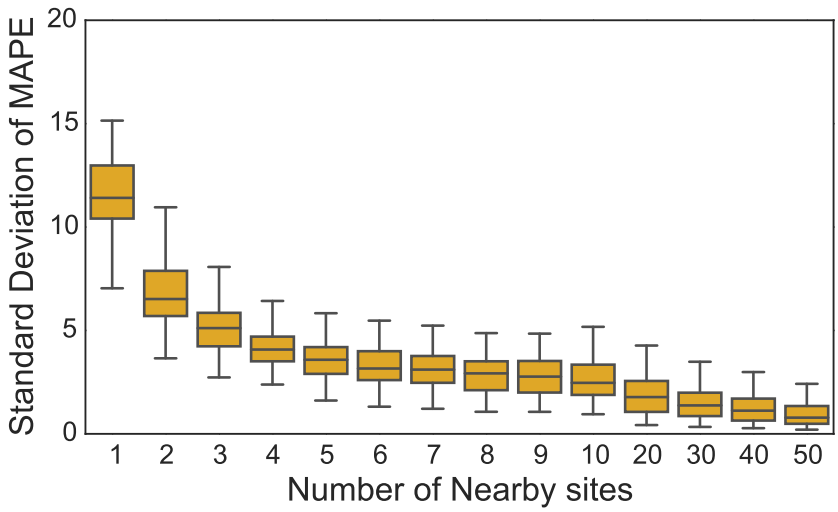


Figure 5.7. Standard deviation of MAPE diminishes with increase in the number of geographically nearby sites.

of the 100 models with increasing number of geographically nearby sites. As shown in the figure, we observe that the variability reduces when the number of nearby sites increases. However, unlike the previous result, the variability continues to reduce — albeit at a slower rate — even when the number of nearby sites is greater than five. Thus, the performance of the learned models is closer to its average.

5.6.3 Detection of anomalies

We illustrate the different steps involved in our algorithm using Figure 5.8. In the top subplot of the figure, the blue line depicts the power generation trace from a solar installation for over a week in August, 2015. The red marker shows the prediction from the RandomForest regression technique with data from the remaining 87 sites as features. While the prediction (i.e., red marker) closely follows the actual power output (i.e., blue line), there is a significant difference in the actual and predicted after 14th August. As seen, there is a sharp drop in the actual power generated in the late morning of 14th August. The drop in power is significant, and there is no output recorded in the site for an extended

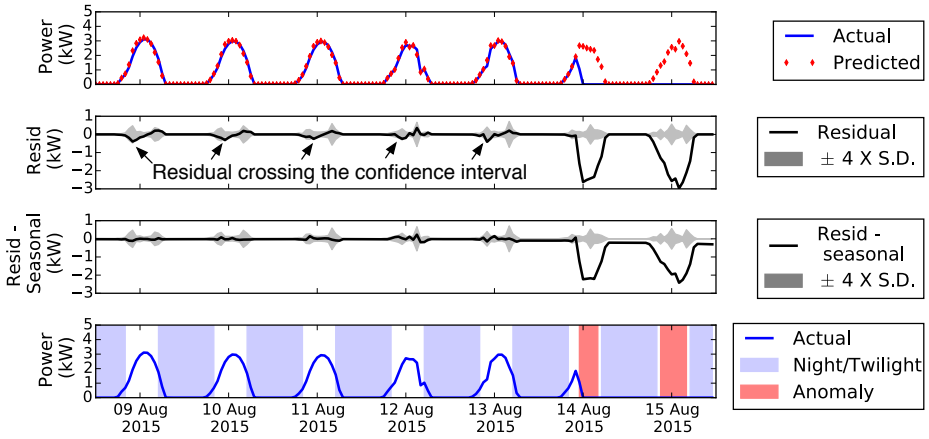


Figure 5.8. An illustrative example that depicts the data-processing and anomaly detection steps in SolarClique.

period until October (not shown in the figure). However, the regression model forecasts a non-negative power output for the given site.

The second subplot shows the residual, i.e., the difference between the actual and the predicted values (i.e., the black line) along with the confidence interval (i.e., the gray shaded region). The confidence interval, which is within $\pm 4\sigma$, is calculated using the pointwise standard deviation obtained from the bootstrap process. In this figure, we observe that the residual sometimes lie outside the confidence interval at the same time of the day across multiple days — which indicates a fixed periodic component.

On removing the seasonal component using our approach, we observe that the residual always lies within the confidence interval, except when there is an anomaly in power generation. This is shown in the third subplot of the figure, where the black line (i.e., residual) lie within the gray shaded region (i.e., the confidence interval). Finally, the last subplot depicts our anomaly detection algorithm in action. We observe that our algorithm accurately flags periods of no output as an anomaly (depicted by the red shaded region).

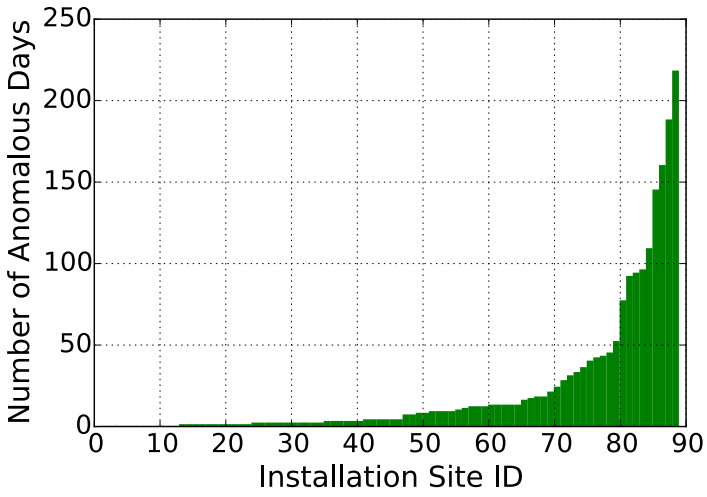


Figure 5.9. Number of anomalous days for each site. Installation sites are plotted in ascending order of anomalous days.

5.7 Case-study: Anomaly Detection Analysis

In this case study, we use the solar installations in the Dataport as they represent a typical setup within a city. We ran our SolarClique algorithm on the generation output from all solar installations and obtained the anomalous days in the dataset. Below, we present our analysis.

5.7.1 Anomalies in solar installations

Figure 5.9 shows the total number of anomalous days in each solar installation site. We observe that our SolarClique algorithm found anomalous days in 76 solar installations, out of the 88 sites in the dataset. As seen in the figure, the total number of anomalous days span from a day to several months. Together, all the installation sites had a total of 1906 anomalous days. This indicates a significant loss of renewable power output. Specifically, we observe that 17 of the 88 (around 20%) installations had anomalous power generation for at least a total of one month that represents more than 5% of the overall 640 days in the testing period. Anomalies from these installations account for nearly 80% of all the anomalous days.

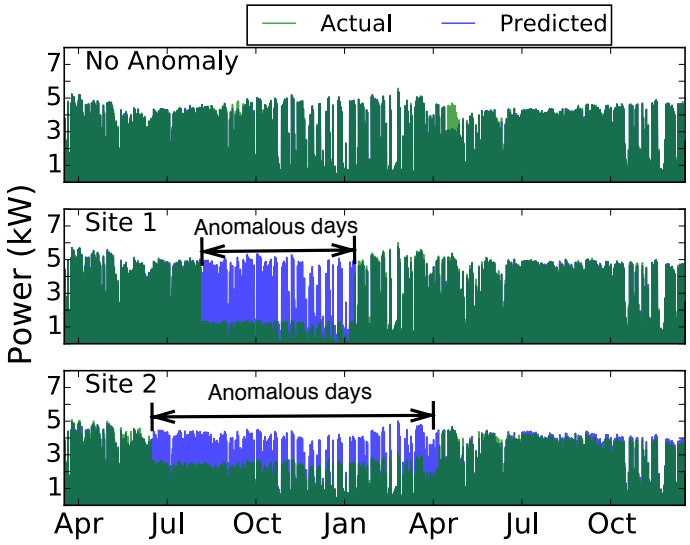


Figure 5.10. Under-production of solar detected using our algorithm.

To better understand the anomalous periods, we group them into short-term and long-term periods. The short-term periods have less than three contiguous anomalous days, while the long-term periods have consecutive anomalous days for at least three days. Our results show the dataset has 587 occurrences of short-term periods spread over 683 days. Further, we observe 123 occurrences of long-term periods spread over 1223 days. We also observe that the maximum contiguous anomalous period found in a site was approximately five months (i.e., 158 days), with no power output during that period. Clearly, such high number of long-term anomalous periods demonstrate the need for early anomaly detection tools. Additionally, we note that long-term anomalies are relatively easier to detect than short-term anomalies. While long-term anomalies represent serious issues that may need immediate attention, short-term anomalies may be minor problems, if unattended, could become major problems in future. The advantage of our approach is we can detect both short-term and long-term anomalies.

5.7.2 Analysis of anomalies detected

Note that the reduction in power output depends on the severity of an anomaly. This is because some electrical faults (e.g., short-circuit of a panel) may have localized impact on a solar array, which can marginally reduce the power output, while other faults (e.g., inverter faults) may show significant power reduction or completely stop power generation.

SolarClique detects anomalous days when there is no solar generation and also when an installation under produces power. Our algorithm reported 1099 and 807 anomalous days with under production and no solar generation, respectively. Since no solar generation days are trivially anomalous, we specifically examine cases of solar under production. Figure 5.10 shows the power output from three different sites. The top plot shows the power output (depicted by the blue line) with no anomalous days, the subplots below show sites that have anomalous days (depicted by the red marker). Our results show that the SolarClique algorithm detects anomalies even when a site under produces solar power. Note that the site with no anomaly, which is exposed to the same solar irradiance as other sites, continues to produce solar output. However, we observe a drop in power output for an extended period in the anomalous sites. Specifically, we observe the drop in power output is around 75% and 40% in Site 1 and Site 2, respectively — presumably due to factors such as line faults in the solar installation. Usually, anomalies such as line faults can cause a significant drop in the power output. In particular, a 75% drop in Site 1 can be attributed to faults in three fourth of the strings (i.e., connected in series).

We further examine the reduction in power output in the underproduction cases. Figure 5.11 shows the distribution of the difference in actual and predicted power output for anomalous days. Out of the1099 under production days, our algorithm reported 23 days when the difference in percentage was less than or equal to 5%. Typically, more than 5% drop in power output is considered significant. This is because malfunctioning of a single

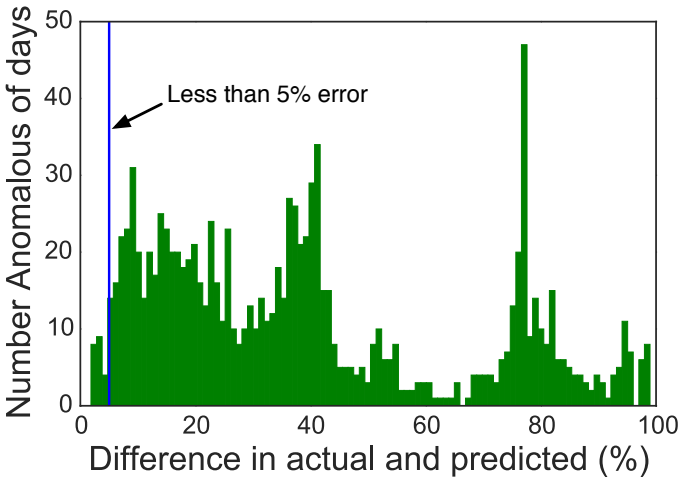
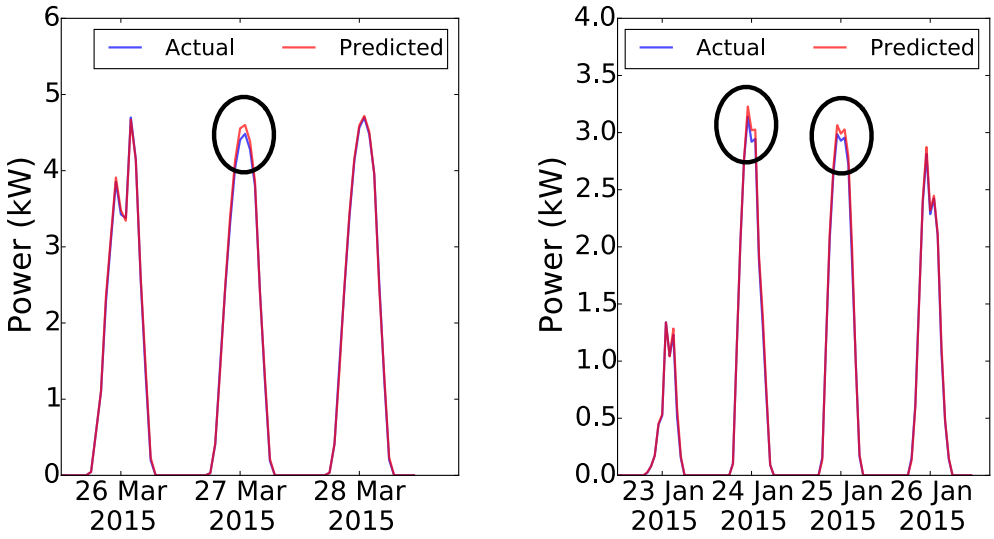


Figure 5.11. Distribution of the difference in actual and predicted on underproducing anomalous days.

panel in a solar array with 20 panels[2] will result in a 5% reduction. Thus, we investigate anomalous days wherein the difference is less than or equal to 5%. Figure 5.12 compares the regression fit of anomalous days with two normal days (adjacent to the anomalous days) from two sample sites where the difference was less than 5%. Note that the figure shows a good fit for most periods except during the anomalous period highlighted in the circle. In comparison to other periods, we observe a drop in power during the anomalous period, occurring during the mid-day. Even though the difference in percentage is small, it represents a relatively significant drop since the power output is at its peak during the mid-day.

We observe that our approach also detects anomalies due to degradation in the power output, which usually spans over an extended time period. Since the drop in power output over the time period may be small, such changes are more subtle and harder to identify. Figure 5.13 shows the degradation in power output of an anomalous site. Our algorithm reports an increase in the frequency of anomalous days in the installation site over the year, with more anomalous days in the latter half. To understand the increase in anomalous days, we plot the difference between the actual and predicted (seen in the bottom subplot). We

[2]Typically, a 5kW installation capacity has 20 panels, each panel having 250W capacity.



(a) Site A (b) Site B

Figure 5.12. Anomalies detected in two sample sites where the difference in actual and predicted was less than 5%. The figure shows a good fit on all days except the anomalous period highlighted in the circle.

observe that the difference between the actual and predicted value steadily increases over time. It is known that the power output of solar installations may reduce over time due to aging [77] at a rate of around 1% a year. However, the accelerated degradation seen in Figure 5.13 is presumably due to occurrences of hot-spots or increased contact resistance due to corrosion. Early detection of such conditions can help homeowners take advantage of product warranties available on solar panels.

We now examine the types of anomalies in the top 17 sites with more than a month of anomalous days. The power output of anomalous days can be categorized into three types — (i) no production, (ii) under production, and (iii) degradation over a period. Table 5.2 summarizes the different types occurring over a period in these sites. The single period represents a single contiguous period of anomaly, while the multiple period represents more than one contiguous period. We observe that the average power reduction during anomalous periods may range from 98.8% to 30.6%. We classify "no production days" as days with no power output for the majority of the period. Overall, we observe that there are 810 no

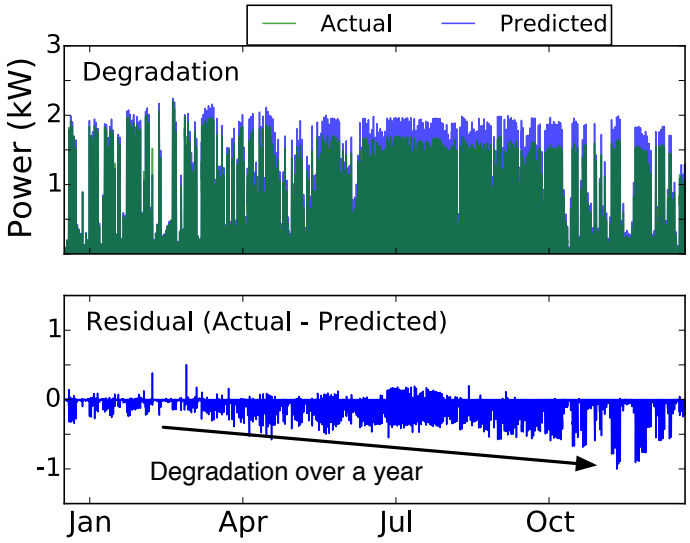


Figure 5.13. Accelerated degradation in the power output of a solar site.

production days — a significant loss in renewable output. Although the average power reduction due to severe degradation is 30%, it is likely to grow over time.

5.8 Discussion

As mentioned earlier, several third-party sites exist that host solar generation data for rooftop installations. While in our approach, we use power to determine the existence of anomalies in power generation, several other electrical characteristics such as voltage and current are available that carry much richer information about the type of anomaly. This information can be leveraged to further infer the exact type of anomaly in power generation. For example, a line fault (broken string) will reduce the current produced by the overall setup, but the voltage will remain unchanged. Conversely, covering of dust/bird droppings can impact both the voltage and the current. Thus, our algorithm can be extended to use multi-modal data (e.g., voltage, current, and power) to further diagnose the exact cause of the anomaly.

Anomaly Type	**#Sites**	**#Days**	**Avg. power reduction(%)**
Single No Production	5	515	98.87
Multiple No Production	3	295	98.65
Single Under Production	2	348	60.22
Multiple Under Production	4	164	43.63
Severe Degradation	3	179	30.67

Table 5.2. Types of anomaly in sites having more than a month of anomalous days.

Our approach can also be extended to a single solar installation for detecting anomalies. With the proliferation of micro-inverters in residential solar installations, power generation data from individual panels are available. Power output from these colocated panels can also be used to detect faults in the PV setup, as they can predict the power output with higher fidelity. This can be used in remote locations where data from other solar installations are not easily available. As part of future work, we plan to use SolarClique algorithm to discover faults in a single panel by comparing power generated with others in the same setup.

5.9 Related Work

There has been significant work on predicting the solar output from solar arrays [14, 15, 53, 71, 94]. While some studies have used site-specific data such as panel configuration [15, 71] for building the prediction model, others have used external data such as weather or historical generation data [58, 94]. Such models can provide short-term generation forecast (e.g., an hour) to long-term forecast (e.g., days or weeks). Although these studies can predict the reduction in power output, a limitation in these studies is that they cannot attribute the reduction to anomalies in the solar installation.

Prior work has also focused on anomaly detection in PV panels [45, 46, 77, 88, 97, 99, 101]. These studies propose methods to model the effects of shades/covering [45, 65], hot-spots [62], degradation [77, 97] or short-circuit and other faults [46]. However, these methods require extensive data (such as statistics on different types of anomalies) [96] or do not focus on hardware-related issues [45]. For instance, [96] proposes a solution to determine probable causes of anomalies but require detailed site-specific information along with pre-defined profiles of anomalies. Unlike prior approaches, our approach doesn't require such extensive data or setup and relies instead on power generation from co-located sites. Thus, it provides a scalable and cost-effective approach to detect anomalies in thousands of solar installation sites.

The idea behind our approach is similar to [89, 90]. However, the authors use the approach in the context of an astronomy application, wherein systematic errors are removed to detect exoplanets. In this case, the systematic errors are confounding factors due to telescope and spacecraft, which influences the observations from distant stars. In contrast, our solution uses inputs from other geographically nearby sites to detect anomalies in solar. As discussed earlier, today, such datasets are easily accessible over the internet, which makes our approach feasible. Further, using regression on the data from neighbors has been studied earlier [32]. However, the main focus of this work was in the context of quality control in climate observations by imputing missing values. In our case, we use the learned regression model to find anomalous solar generation.

5.10 Conclusion

In this chapter, we proposed SolarClique, a data-driven approach to detect anomalies in power generation of a solar installation. Our approach requires only power generation data from geographically nearby sites and doesn't rely on expensive instrumentation or other external data. We evaluated SolarClique on the power generation data over a period of two years from 88 solar installations in Austin, Texas. We showed how our solar installation

regression models are accurate with tight confidence intervals. Further, we showed that our approach could generate models with as few as just five geographically nearby sites. We observed that out of the 88 solar installations, 76 deployments had anomalies in power generation. Additionally, we found that our approach is powerful enough to distinguish between reduction in power output due to anomalies and other factors (such as cloudy conditions). Finally, we presented a detailed analysis of the different anomalies observed in our dataset.

CHAPTER 6

MODEL-DRIVEN ENERGY EFFICIENCY ANALYTICS AT CITY-SCALE

This chapter presents WattHome, an algorithm that utilizes at grid-level smart meter data to initially derive a probabilistic and weather-sensitive energy model for individual homes based on its energy usage and the prevailing ambient weather. The model parameters are then leveraged to identify the least efficient buildings in a given population along with the underlying cause of energy inefficiency. This chapter also discusses a detailed evaluation section that examines the individual parts of the algorithm. Finally, the chapter presents a real-world case-study where WattHome is used to identify faults in residential buildings in a mid-sized city in the New England region.

6.1 Motivation

A city consists of hundreds or thousands of buildings, an essential first step for implementing energy-efficiency measures is to identify those that are the least efficient and thus have the greatest need for energy-efficiency improvements. Interestingly, naive approaches such as using the age of the building or its total energy bill to identify inefficient buildings do not work well. While older buildings are usually less efficient than newer ones, age alone is not an accurate indicator of efficiency, since older buildings may have undergone renovations and energy improvements. Similarly, the total energy usage is not directly correlated to energy inefficiency. First, larger buildings will consume more energy than smaller ones. Even normalizing for size, greater energy usage does not necessarily point to inefficiencies. For example, two identical size homes with a different number of

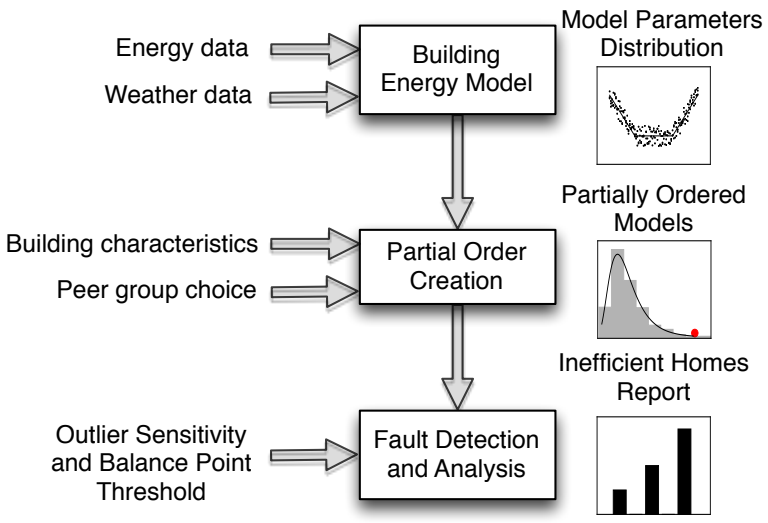


Figure 6.1. WattHome Overview.

residents will have different total energy usage, and higher usage, in this case, merely reflects a greater number of occupants rather than inefficiency. Thus, finding truly inefficient buildings requires more sophisticated methods.

The buildings that are identified as inefficient can then become candidates for various energy efficiency measures such as energy audits or targeted energy incentives for improvements or upgrades. Methods to identify inefficient homes is made feasible by the availability of citywide datasets. For example, advanced metering infrastructure in smart grids, also known as smart meters, can monitor a buildings energy usage at a fine time granularity of minutes or hours. Real estate information describing a building's age, size, and other characteristic are public records in many countries, and curated building datasets for entire cities is readily available through public APIs over the Internet.

6.2 WattHome Approach

In this section, we describe the details of our data-driven approach. WattHome's approach is depicted in Figure 6.1 and it involves three key steps: (i) Learn a *building energy model* for each home from energy usage data, (ii) Create a *partial order* of buildings using

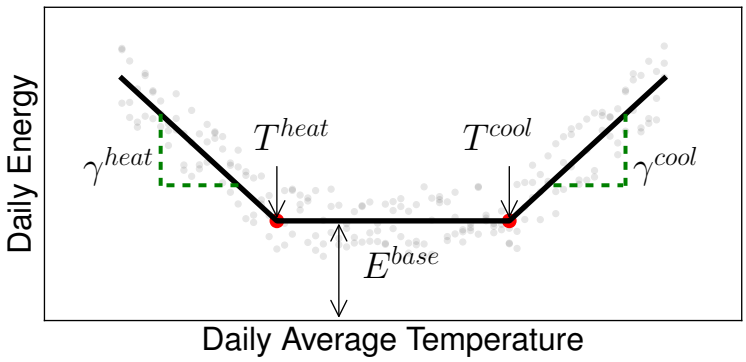


Figure 6.2. Energy usage versus outdoor temperature.

its parameter distribution from the building model, and finally (iii) Detect *building faults* causing energy inefficiency. Below, we discuss each step in detail.

6.2.1 Building Energy Model

We first provide the intuition behind our approach. Heating and cooling costs for a building can be understood using elementary thermodynamics. Typically, in colder months, the outside ambient temperature is colder than the inside building temperature, resulting in a net thermal loss where the inside heat flows outside through the building envelope, causing the inside temperature to drop. In warmer months, the opposite is true. The building experiences a net heat gain where the heat flows inside, causing the building temperature to rise.

It follows that every home has a specific temperature T_b, where there is neither thermal loss nor thermal gain i.e. the thermodynamic equilibrium. When the outside temperature is above T_b, there is a need for AC to cool the home. Conversely, when the temperature is below T_b, there is a need for a heater to heat the home. This temperature T_b is called the balance point temperature of the building. The rate of thermal loss or thermal gain depends on the degree of insulation, airtightness of the building envelope and surface area exposed to outside elements. Better the insulation and airtightness, smaller the rate of loss or gain for a given temperature differential relative to T_b. The difference between the

outside temperature and the balance point temperature T_b is also referred as the *degree-days* — an indication of how many degrees warmer or colder is the outside weather relative to the building's balance point.

Based on this intuition, we now describe our building energy model. Any energy load in a building can be classified as weather independent and dependent. A weather independent load is one where the energy consumed by the device is uncorrelated to the outside temperature — consumption from loads such as lighting, electronic devices, and household appliances depend on human activity rather than outside weather. Heating and cooling equipment constitute weather dependent loads, as their consumption linearly dependent on the outside temperature relative to the balance point.

If we assume that weather independent loads are distributed around a constant value (also called the base load); then the total energy consumed is the sum of the base load and the weather dependent loads (heating and cooling loads) and defined as:

$$E_d^{total} = E_d^{heat} + E_d^{cool} + E^{base} \qquad \forall d \in D \tag{6.1}$$

where E_d^{total} denotes the total energy used by a building on day $d \in D$. E_d^{heat} and E_d^{cool} denote the energy used for heating and cooling, respectively, on day d, while E^{base} denotes the energy usage of base load appliances. Thus, given a series of observations of the total energy usage and the outside ambient temperature, it is possible to fit a regression and learn the fixed weather independent component (base load) and the temperature dependent component (heating and cooling). This forms the basis for inferring our weather-aware building energy model.

Figure 6.2 illustrates the relationship between outdoor temperature and the energy consumption of a building. The individual data points represent the daily energy usage (along the Y-axis) for a given average outdoor temperature (along the X-axis) of a building. The figure shows that the building has two balance point temperatures — a heating balance point temperature T^{heat}, below which heating units are turned on, and a cooling balance

point temperature T^{cool}, above which air-conditioning is turned on. Further, the figure also shows a piecewise linear fit over the daily energy usage. When the outdoor temperature is between the two balance points, the building consumes energy that is distributed around a constant value defined as the *base load* E^{base} energy consumption. The weather dependent components, i.e. the heating E^{heat} and cooling E^{cool} energy consumption, are a function of ambient outdoor temperature T_d and are defined as:

$$E_d^{heat} = \gamma^{heat}(T^{heat} - T_d)^+ \quad \forall d \in D \tag{6.2}$$

$$E_d^{cool} = \gamma^{cool}(T_d - T^{cool})^+ \quad \forall d \in D \tag{6.3}$$

where γ^{heat} and γ^{cool} are the heating and the cooling slope in the above linear equations and represent a positive constant factor indicating the sensitivity of the building to temperature changes; and $()^+$ indicates the value is zero if negative and ensures either energy from heating or cooling is considered. Using (6.2) and (6.3), energy model in (6.1) can be represented as a piecewise linear model:

$$E_d^{total} = E^{base} + \gamma^{heat}(T^{heat} - T_d)^+ + \gamma^{cool}(T_d - T^{cool})^+ \forall d \in D \tag{6.4}$$

The model in (6.4) is known as the *degree-day* model [63] and forms our base energy model for estimating the building parameters.

While methods like Maximum Likelihood Estimation (MLE) or Maximum a posteriori estimation (MAP) can be used for determining the building parameters, they provide point estimates that can hide relevant information (such as not capturing the uncertainties in human energy usage). To capture human variations, we require probability density function of the parameters. Thus, we use Bayesian inference approach, which provides the posterior distribution of parameters.

We model (6.4) using a bayesian approach and assume the error process to be normally distributed ($\mathcal{N}(0, \sigma^2)$). Thus, the daily energy consumption E_d^{total} is normally distributed with parameters mean (μ) and variance (σ^2), where μ is equal to the right hand side of (6.4). Note that energy consumption E_d^{total} is known and so is the independent variable i.e. ambient temperature T_d. However, the building parameters (γ^{heat}, γ^{cool}, T^{heat}, T^{cool}, and E^{base}) are unknown. Using Bayesian inference, we can then compute a *posterior* distribution for each of these parameters that best explains the *evidence* (i.e. the known values for E_d^{total} and $T_d \forall d \in D$) from initially assuming a *prior* distribution.

To determine the posterior distribution of the individual parameters, we use the Markov chain Monte Carlo (MCMC) method that generates samples from the posterior distribution by forming a reversible Markov-chain with the same equilibrium distribution. We introduce a prior distribution that represents the initial belief regarding the building parameters. For example, the two balance point temperatures will be between a wide range of 32°F and 100°F. This belief can be represented using a uniform prior with the said range. Similarly, the baseload, heating slope and cooling slope can be drawn from a weakly informative gaussian prior having non-zero values. This is because baseload, a unit of energy, cannot be negative. Similarly, slope values must be positive as they represent increase in energy per unit temperature. The parameters of the gaussian priors are scaled to our setting and selected based on the recommendations provided by Gelman et al. [48]. To simplify our building model, we assume that the parameters are independent, i.e., the heating, cooling and the baseload parameters do not affect one another.

Several MCMC methods leverage different strategies to lead from these priors towards the target posterior distribution. We employed No-U-turn sampler, a sophisticated MCMC method, which has shown to converge quickly towards the target distribution. Thus, after an initial *burn in* samples, we can draw samples approximating the true posterior distribution. From these post-burn-in samples, a posterior distribution for the individual building parameters can be formed. Our complete Bayesian model is defined in Table 6.1.

Prior

$E^{base} \sim \mathcal{N}(20, 20)$, $\gamma^{heat} \sim \mathcal{N}(0, 4)$, $\gamma^{cool} \sim \mathcal{N}(0, 4)$

$T^{heat} \sim \mathcal{U}(32, 100)$, $T^{cool} \sim \mathcal{U}(32, 100)$, $\sigma \sim Cauchy(0, 5)$

Regression Equation

$\mu_d = E^{base} + \gamma^{heat}(T^{heat} - T_d)^+ + \gamma^{cool}(T_d - T^{cool})^+ \; \forall d \in D$

Model Likelihood

$E_d^{total} \sim \mathcal{N}(\mu_d, \sigma^2)$

Parameter Bounds

$E^{base}, \gamma^{heat}, \gamma^{cool} >= 0 \quad \text{and} \quad T^{heat} <= T^{cool}$

Table 6.1. Bayesian formulation of our building energy model.

Since buildings are of different sizes, simply comparing the parameters in absolute terms is not meaningful. To enable such comparison, we initially normalize the energy usage by building size before the Bayesian inference. Hence, in our case, E^{base} represents base load energy use per unit area. Similarly, heating slope γ^{heat} and cooling slope γ^{cool} gives change in energy per degree temperature per unit area. Thus, the balance point parameters (T^{heat} and T^{cool}) are not normalized as they are unaffected by the size of the house. We construct a cumulative distribution ($F_{\gamma^{heat}}$, $F_{\gamma^{cool}}$, $F_{E^{base}}$) for each of the building model parameter (γ^{heat}, γ^{cool}, E^{base}) from their respective density functions (posterior) obtained after the inference. For the balance point parameters (T^{heat} and T^{cool}), we only use its mean values as they tend to remain fixed for a given building irrespective of human variation. This completes our approach for creating the building energy model.

6.2.2 Partial Order Creation

Rather than relying on rule-of-thumb measures to interpret model parameters that change with geography and many other building characteristics, we propose comparing

them with those of similar homes from a given population. Given the above model, we create a partial order of buildings as follows. We first create *peer groups* using the building's physical attributes (e.g., age of the building, building type etc.). Next, within each peer group we create a *partial order* of the buildings for each building parameter distribution. Below, we describe each step in detail.

- **Peer groups creation:** To enable a meaningful comparison, we compare the building model parameters only within their cohort. We use three building attributes for peer group creation namely: (i) property class (e.g., single family, apartment, etc.), (ii) built area (e.g., 2000 to 300 sq.ft.), and (iii) year built (e.g.1945 to 1965). For instance, buildings constructed in different years adhere to different energy regulations and standards, and thus, it is not meaningful to compare them. Similarly, building types and age group have different characteristics and it would be unreasonable to compare them. Hence, our approach allows the creation of peer groups to enable comparison within a cohort to determine inefficient homes.

- **Stochastic order of building parameters:** Since the building model parameters are probabilistic distributions, we cannot simply compare these uncertain quantities and create a *total ordering*. Statistics, such as mean, median or mode, provide a single number to capture the behavior of the whole distribution. While these *point estimates* can be used to compare two distributions, they typically hide useful information regarding their shape and may not account for any heavy-tailed nature that is present in a building parameter distribution. Hence, we use *second order stochastic dominance*, a well-known concept in decision theory for comparing two distributions [68], to create a partial order of the building parameters within a peer group.

  The main idea behind determining *second order stochastic dominance* is that for a given building model parameter p, if distribution F_p dominates G_p i.e., $F_p \succeq_2 G_p$,

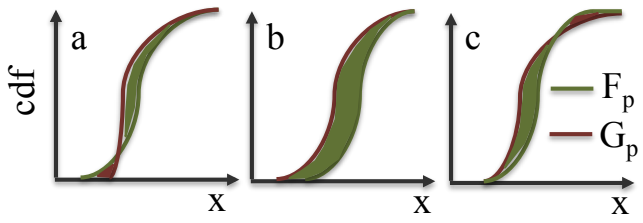


Figure 6.3. Stochastic ordering of two distributions F_p and G_p. (a) F_p does not dominate G_p. In (b) and (c) F_p dominates G_p.

then the area enclosed between F_p and G_p distribution should be non-negative up to every point in x:

$$\int_a^x (G_p(t) - F_p(t))dt \geq 0 \qquad \forall x \in [a, b] \tag{6.5}$$

Figure 6.3 depicts stochastic ordering of two distribution F_p and G_p where; (i) F_p does not dominate G_p i.e. $F_p \not\succeq_2 G_p$ and (ii) F_p dominates G_p i.e., $F_p \succeq_2 G_p$. The area shaded in green shows the region where F_p dominates G_p, and the red region shows G_p dominates F_p. In Figure 6.3(a), we observe that $F_p \not\succeq_2 G_p$, since there are no green area greater or equal to the left of the red area. In contrast, Figure 6.3(b) and (c) shows F_p dominates G_p because for every red area, there exists a larger green area located to its left.

To intuitively understand the implications of stochastic dominance in our scenario, let us consider two distributions F_p and G_p of a building parameter p from two separate buildings A and B respectively. As noted earlier, building parameters influences energy usage, such that higher parameter values implies higher energy usage, and vice-versa. Let us assume that building A's normalized energy usage is greater than building G's normalized energy usage, such that distribution F_p dominates G_p i.e., $F_p \succeq G_p$. Clearly, the building parameter distribution F_p for building A will lie on the right-side of distribution G_p as A has higher energy usage. In fact, since $F_p \succeq G_p$,

Indicator Characteristics	Probable Building Faults
High Heating Slope	Inefficient Heater, Building Envelope
High Cooling Slope	Inefficient AC, Building Envelope
High Heating Balance Point	High Set point, Poor Building Envelope
Low Cooling Balance Point	Low Set point, Poor Building Envelope
High Base load	Inefficient Appliances

Table 6.2. Indicator building model characteristics and associated probable building faults.

by definition, the distribution F_p will be on the right of G_p for a majority of the region. However, homes may have similar building parameter distribution, i.e the distribution has similar shape and tendency. In such cases, it is possible that neither home will dominate the other. Stochastic dominance thus enables interpretation of the building parameter distribution with respect to one another, with higher energy usage buildings having a tendency to lie on the right side of the population. This allows separation of homes with dominant distributions from non-dominant ones. We run a pair-wise comparison of all buildings within a cohort for each building model parameter p. This gives us the partial order for all pairs and parameters, which we use to detect inefficient homes.

6.2.3 Fault Detection and Analysis

We first discuss the causes of inefficiencies associated with the different model parameters. Heating slope γ_{heat} and heating balance point temperature T^{heat} are the two parameters that enable our model to interpret the heating inefficiencies of a home. Buildings with high γ^{heat} lose heat at a higher rate, which in turn affects heating unit usage (i.e., consumes more power) to compensate for the high loss rate. A high energy loss rate can be attributed to poor building insulation, air leakages, or inefficient or heating unit. Separately, heating

balance point temperature also indicates inefficiencies in the heating component of a home. A high balance point temperature suggests two possible inefficiencies: (i) high thermostat set-point temperature[1] and (ii) poor building insulation. If the set-point temperature is high during winters, heating units turn on more frequently to maintain the indoor temperature at set-point. In contrast, if building insulation is poor, more heat is lost through the building envelope. Thus, heating units will be turned on frequently to sustain the high heating balance point temperature. Similarly, we can interpret the cooling slopes γ^{cool} and cooling balance point temperature, which points to inefficiencies in cooling units or building envelope.

Homes with high E^{base} indicate high appliance usage or inefficient appliances. In such homes, energy retrofits may not help reduce energy consumption. However, these homes may benefit from replacing old appliances (water heater, dryer) with newer energy star rated ones. We summarize the association between probable causes of building faults and model parameter in Table 6.2.

Next, we present our algorithm that identifies inefficient homes and its potential cause. Here, we first use the partially ordered set of buildings to determine the outliers for each parameter and then use the mapping in Table 6.2 to assign building faults. To determine outliers, note that the energy usage of an inefficient home would be high. Thus, the building parameter distribution of an inefficient home will tend to be *stochastically dominant* with respect to others in their peer group. However, among inefficient homes, the building parameter distribution may be similar, and thus their distributions may not be stochastically dominant to one another. Similarly, within energy efficient homes this distinction of dominance may not be apparent, as their distribution may be identical to one another. We use this insight to define a building as *inefficient* in a given model parameter, if it is stochastically dominant compared to a majority of the homes within its cohort. For instance, if

[1]Set point temperature and balance point temperature have a linear relationship

a building's heating parameter distribution $F_{\hat{\gamma}^{heat}}$ is dominant across more than $\tau\%$ of the buildings, we conclude that the building is inefficient and has a *high* heating slope. Here, τ is the sensitivity threshold for WattHome and provides the flexibility to control the number of inefficient homes. The higher the threshold value, the higher the possibility of identifying an inefficient home. For all experiments, we chose this to be 75%. Thus, for each parameter, we determine whether a building is inefficient if its distribution is dominant beyond a certain threshold. We use a balance point threshold to determine buildings with high balance point temperature. We flag buildings as inefficient if the mean value obtained after inference for heating (or cooling) balance point temperature T^{heat} (or T^{cool}) is greater than (less than) specific heating (or cooling) balance point threshold 70°F (55°F) — a common choice employed by expert auditors. We present the pseudo-code to determine inefficient buildings in Algorithm 2.

Algorithm 2 Fault Analysis Algorithm

Inputs: Sensitivity (τ), buildings (B)
procedure FINDINEFFICIENTHOMES(τ, B)
    count = {}; homes = {}
    for p in $[\gamma^{heat}, \gamma^{cool}, E^{base}]$ **do**
        for ($b1$, $b2$) $\leftarrow {}^{|B|}P_2$ **do** // all-pairs permutation
            if $F_p(b1) \succeq_2 F_p(b2)$ **then**
                count[p, $b1$] +=1
        for $b \leftarrow B$ **do** homes[p, b] = count[p, b] $\geq \tau$
    for $b \leftarrow B$ **do** homes[T^{heat}, b] = $T_b^{heat} > 70°F$
    for $b \leftarrow B$ **do** homes[T^{cool}, b] = $T_b^{cool} < 55°F$
    return homes

Inputs: building (b), parameters (P), fault_map (M)
procedure GETROOTCAUSE(h, P, M)
    faults = []
    for $p \leftarrow P$ **do**
        if homes[p, b] **then**
            faults += $M[p]$ // append list
    return faults

As noted earlier, each parameter in the building model affects an energy component defined in (6.4). Any irregularity in the building parameter, in comparison to its peer group,

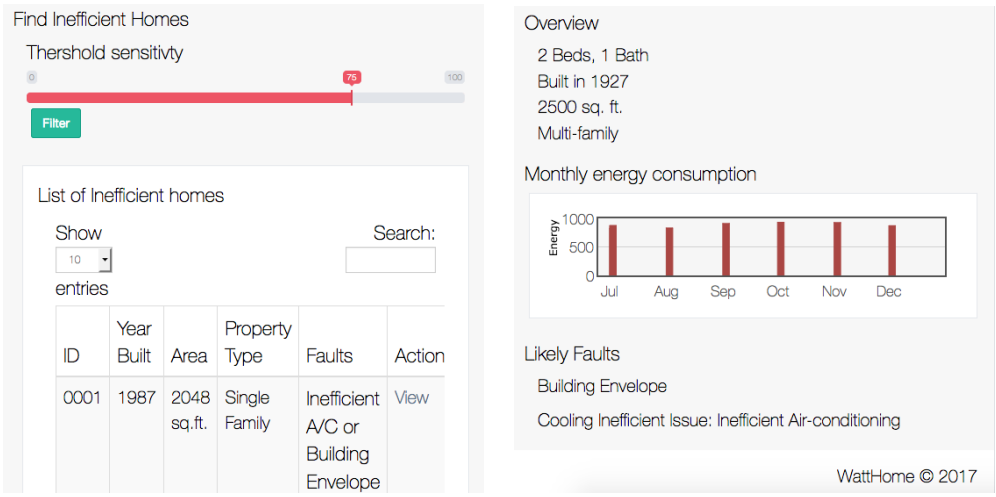
points to possible inefficiency in the said energy component. We outline our pseudo-code for finding root cause in Algorithm 2. First, we create a mapping of indicators of deviations in building model parameters to possible faults using Table 6.2. We provide the mapping as an input to our algorithm. Next, we associate a fault to a home if it was flagged inefficient for the given parameter p. For instance, if a home is flagged as high base load, we say that the home has inefficient appliances. Similarly, an inefficient home with high heating slope is assigned faults related to heating inefficiencies. We then generate a report of the list of potential faults in a given home.

6.3 Implementation

We implemented WattHome as an open source tool. WattHome is split into two components — (i) a Unix-like command line tool [2] that uses PyStan, a statistical modeling library, to implement our bayesian model, and (ii) a web-based application interface implemented using Django framework for interacting with the command line tool. Users can interact with either component, and provide their energy traces and building information, to determine likely reasons of inefficiency.

Our system works as follows. When users provide their energy traces and building information (such as zip code, year built, etc.), WattHome builds a custom bayesian model of the home using the local weather data and the details provided by the user. The weather data of a nearby airport is used as a proxy for local weather conditions, and WattHome periodically fetches and updates this data from public APIs. Next, users provide a sensitivity threshold that is used to create a partially ordered set of inefficient homes. As utility companies may have a limited audit budget to manually inspect homes, the threshold provides user the flexibility to control the list of least efficient home. Figure 6.4(a) shows how users can adjust the sensitivity parameter to get inefficient homes. Finally, our WattHome

[2]We have publicly released the code and the tool. `http://bit.ly/2nU7kA5`



(a) Find Inefficient homes (b) Inefficiency Report

Figure 6.4. Screenshot of our implementation of WattHome.

generates a report listing inefficient homes and their likely faults. Figure 6.4(b) shows the inefficiency report for a single home listing likely faults.

6.4 Experimental Validation

We first validate our model estimates against ground truth data from two cities and evaluate its efficacy.

6.4.1 Dataset Description

- **Dataset 1: Dataport (Austin, Texas) :**

  Our first dataset contains energy consumption information from homes located in Austin, Texas from the Dataport Research Program [4]. The dataset contains energy breakdown at an appliance level, which serves as ground truth to understand how our approach disaggregates energy components. We select a subset of homes (163 in total) from this dataset having HVAC, baseload appliances along withthe total energy usage information. Since most homes enrolled in the Dataport research program are

Charactersitics	Dataset 1	Dataset 2
# of Homes	163	10,107
Duration	2013	2015
Built Area Range (sq.ft.)	758-6516	250-10,000
Year Built Range	1912-2014	1760-2013
Location	Austin, TX	A city in New England

Table 6.3. Key characteristics of Dataport and New England-based utility smart meter dataset

energy-conscious homeowners, and have energy efficient homes, we use this dataset only for validating our energy disaggregation process.

- **Dataset 2: Utility smart meter data (New England):**

  This dataset contains smart meter data for 10,107 homes from a small city in the New England region of the United States [57]. The dataset has energy usage (in kWh) from both electricity and gas meters. Each home may have more than one smart meter — such as a meter to report gas usage and another to report electricity usage. For homes with multiple meters (gas and electric), we combine their energy usage to determine the building's daily energy consumption for an entire year (2015). Apart from energy usage, the dataset also contains real estate information that includes building's size, the number of rooms, bedrooms, property type (single family, apartment, etc.). We also have manual audit reports for some of the homes. We use this as our ground truth data for validating our approach. Further, we have weather information of the city containing average daily outdoor temperature. We summarize the characteristics of both the datasets in Table 6.3.

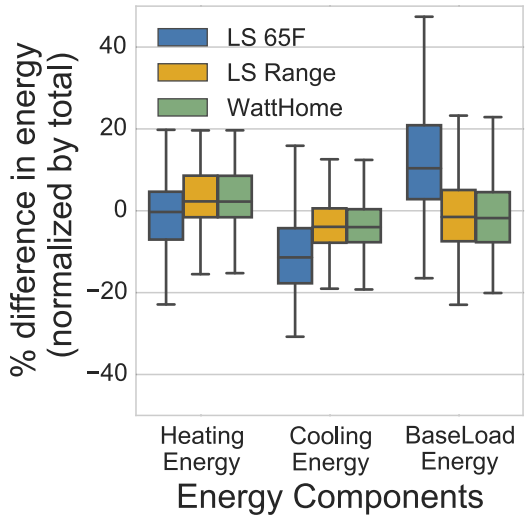


Figure 6.5. Validation of energy split using the two baselines and our model.

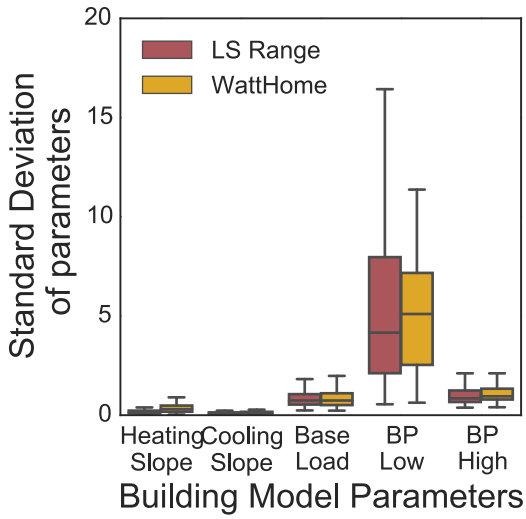


Figure 6.6. Comparison of the standard deviation of parameters.

6.4.2 Energy Split Validation

We now validate the efficacy of our model in disaggregating the overall energy usage into distinct energy components, i.e., heating, cooling, and baseload. For this experiment, we restrict our analysis to the 163 homes from the Dataport dataset.

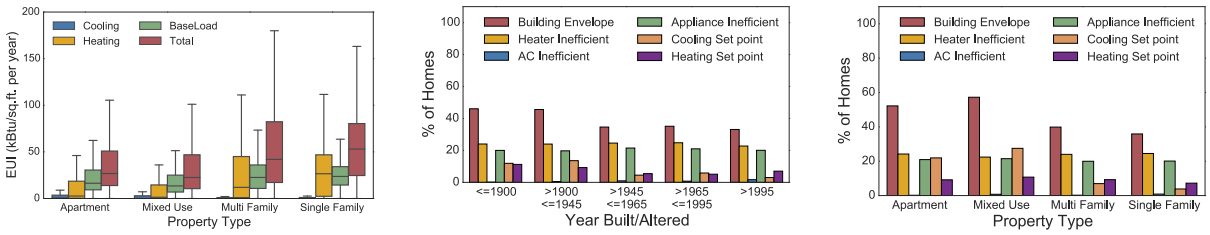
We compare our technique with two baseline techniques (*LS 65F* and *LS Range*), commonly used in prior work, which use the degree-days model to provide point estimates of the individual building model parameters. Our first baseline technique, *LS 65F*, estimates the three building energy parameters (γ^{heat}, γ^{cool}, σ, E^{base}) using least-squares fit and assumes the balance point temperature to be constant (65°F). This is a widely used approach by energy practitioners around the US and recommended by official bodies such as ASHRAE [13]. Our second baseline technique, *LS Range*, estimates all the five building energy parameters (γ^{heat}, γ^{cool}, T^{heat}, T^{cool}, and E^{base}) using the least-squares fit. Unlike the baseline approaches, WattHome estimates the parameter distribution and thus to compare we use the mean of the posterior distribution of the parameters to get the fixed proportion of the energy splits.

Figure 6.5 shows the distribution of percentage difference in the energy usage with the ground truth for each energy component. While *LS Range* and *WattHome* have median error of ≈-1.6%, *LS 65F* have a median error of 10% for baseload energy. Unlike *LS 65F*, *LS Range* and *WattHome* do not assume a constant balance point temperature and thus have lower error. Figure 6.6 compares the standard deviation of the building parameters from the two approaches. In *WattHome*, the standard deviations are obtained from the parameter posterior distributions. Whereas, in case of *LS Range*, the standard deviations are calculated from the covariance matrix outputted by the least-squares routine. While the results for the four parameters are similar, the spread of standard deviation for the lower balance point is much smaller in *WattHome* compared to *LS Range*. In summary, fixed parameters provide poor estimate of the building parameter. Further, *WattHome* provides lower error and tighter parameter estimates compared to other baseline techniques (LS Range).

6.4.3 Faulty Homes Validation

We now examine the accuracy of our model in reporting homes with likely faults. We ran our algorithm on all homes in the New England dataset to generate a list of outlier homes for each of the parameter and then compare our results with findings from manual energy audits (ground truth). Since manual audit reports contain faults related to building envelope and HVAC devices only, we only report these results and inefficiencies arising from base energy usage and faulty set points were not analyzed.

To determine the accuracy, we compare an inefficient building's parameter to the audit report conducted in the past and verify whether it has any building faults. The audit reports were manually compiled by an expert on-field auditor identifying and suggesting energy efficiency improvement measures. We find that WattHome reported 59 homes with building envelope faults, out of which 56 buildings were in the audit report, an accuracy of 95%. Moreover, we find that 46 of the 56 homes with building envelope faults also had faulty



(a) Total Energy Split (b) By Building Age (c) By Property Type

Figure 6.7. (a) Disaggregated energy usage for all homes. (b) and (c) Possible fault types in different building groups.

HVAC systems. In summary, *WattHome* identified parameter related faults in a building with high accuracy. In particular, our approach correctly identified 95% of the homes that were flagged by expert auditors as having either faulty building envelope or HVAC systems.

6.5 Case study: Identifying Inefficient Homes In A City

We conduct a case study on the New England dataset to determine the least efficient residential buildings in the city. In particular, we seek to gain insights on the following questions: (i) What percentage of the homes are energy inefficient? (ii) Which groups of homes are the most energy inefficient? (iii) What are the most common causes of energy inefficiency? We first provide a brief analysis of the distribution of the energy split.

6.5.1 Energy Split Distribution Analysis

To get the fixed proportion of the energy split, we use the mean of the posterior estimates to compute the disaggregated energy usage i.e. heating, cooling and base load components. To compare the energy components, we compute the *Energy Usage Intensity* (EUI), by normalizing the energy component with the building's built area. Figure 6.7(a) shows the heating, cooling, base load and total EUI distribution grouped by property type across all homes. The figure shows that the base load is the highest component of energy usage in most Mixed Use and Apartment property types followed by heating and cooling.

Heating Outliers	Cooling Outliers	Base load Outliers	Overall Outliers
3162	1033	2016	5079

Table 6.4. Summary of all inefficient homes in the data set.

However, for Single family homes, the heating cost is usually higher. The high base load can be attributed to lighting, water heating, and other appliances. Further, since the New England region has more winter days, homes require more heating, and thus expected to have a higher heating energy footprint compared to cooling. In particular, the average heating energy required is almost 20$\times$ that of average cooling energy. We also observe that the normalized total energy usage of single and multi family homes is the highest — presumably due to more number of appliances. The median energy EUI of the Single family home is $\approx$53 kBtu/sq.ft. (1 kW=3.412kBtu), which is almost twice that of Apartment homes ($\approx$26.8 kBtu/sq.ft.).

6.5.2 Efficiency Analysis

In this section, we analyze the results of our approach on the utility company's dataset described earlier. We created peer groups to identify inefficient homes in their respective cohort. To do so, we used three building attributes (property type, age, and area), which created 120 peer groups in total. Among these peer groups, we discarded groups with less than 20 homes, as it didn't have enough population size for a meaningful analysis. In all, 67 peer groups containing a total of 186 homes were discarded. Below, we present our analysis on the remaining 9,921 homes.

First, we examine the number of homes that are flagged as inefficient for each of the energy components using our approach. Table 6.4 shows the summary of inefficient homes across all peer groups. We note that a home may have multiple inefficiencies, such as

inefficient heating and high base load and thus may be inefficient in several of the energy components. Our results show that the overall percentage of inefficient homes across all residential homes is 50.25%. Further, almost 62.25% of all inefficient homes have either inefficient heater or poor building envelope, and 4144 homes have either inefficient heating or cooling.

We now analyze the cause for inefficiency in these inefficient homes. Figure 6.7(b) shows the percentage of inefficient homes within each building age group across all faults. Note that a home may have multiple faults. We observe that the building envelope fault is the major cause of inefficiency, followed by inefficiency in heaters and other base load appliances. Across all age groups, nearly 41% of the homes have building envelope faults, while 23.73% and 0.51% homes have heating and cooling system faults respectively. The figure also shows that some homes might have set point faults. In particular, 18.06% of the homes have issues with either high heating or low cooling set point temperature. These faults indicate likely issues with thermostat setting. Adjusting the thermostat set point temperature in these home may likely improve its efficiency. As shown, homes built/altered before 1945 have a higher proportion of inefficient homes. However, the percentage difference with other age groups is $<$15%.

Figure 6.7(c) shows the percentage of inefficient homes within each building property type and faults. We observe that the building envelope faults are the most common faults across all building types. Further, we find that except for HVAC appliance related faults, mixed use property type has the highest percentage of inefficiency in the remaining fault categories. After mixed use property type, apartments tend to have a higher percentage of inefficient homes followed by multi family and single family property types.

6.6 Related Work

Diagnosing and reducing energy consumption in buildings is an important problem [22, 42, 60, 107]. Various methods have been proposed to detect abnormal energy consumption

in a building [39, 60, 91]. However, these methods focused on commercial buildings that require expensive building management systems [39,91] or requires costly instrumentation using sensors for monitoring purposes [22, 59]. Sensors allow fine-grained monitoring of energy usage but are not scalable due to high installation costs. Unlike prior approaches, our model does not require building management systems or costly instrumentation and use ubiquitous smart meter data to determine energy inefficiency in buildings.

Prior work have also proposed automatic modeling of residential loads [8]. Studies have shown that compound loads can be disaggregated into basic load patterns. Separately, there has been studies on non-intrusive load monitoring (NILM), which allow disaggregation of a household's total energy into its contributing appliances, and does not require building instrumentation [21, 50]. However, most NILM techniques require fine-grained datasets for training purposes and assume energy consumption patterns are similar across homes [21]. On the other hand, our approach makes no such assumption on energy consumption patterns and is applicable across multiple homes as it uses coarse-grained energy usage data that are readily available from utility companies [6].

Various energy performance assessment methods exist to quantify energy use in buildings and identify energy inefficiency [54, 100, 103]. A common approach is to use degree-days method, a linear regression model, for calculating building energy consumption [40, 41, 63]. However, these approaches do not consider uncertainties that are associated with indicators of building performance. The idea of modeling uncertainties in thermal comfort is studied in [33]. However, it is restricted to a single office building with cooling and heating systems. Unlike previous studies, our approach can be used to identify least energy efficient home at scale without manual expert intervention. Further, we propose a novel Bayesian model to account for uncertainties arising from human factors. Finally, we use actual ground truth data to validate our approach and show its efficacy on a large scale city-wide data.

6.7 Conclusions

Improving efficiency of buildings is an important problem, and the first step is to identify inefficient buildings. In this chapter, we proposed WattHome, a data-drive approach to identify the least energy efficient homes in a city or region. We also implemented our approach as an open source tool, which we used to evaluate datasets from different geographical locations. We validated our approach on ground truth data and showed that our model correctly identified 95% of the homes with inefficiencies. Our case study on a city-scale dataset showed that more than half of the buildings in our dataset are energy inefficient in one way or another, of which almost 62.25% of homes with heating related inefficiencies as probable cause. This shows that a lot of buildings can benefit from energy efficiency improvements.

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Conclusions

This thesis explored the opportunities and challenges in leveraging data-driven modeling to smart buildings. We proposed using data from IoT sensors to generate actionable insights for better energy management at different levels of human energy consumption.

In **Chapter 3**, we looked at the usage and energy consumption profiles of residential electric loads. I proposed Non-Intrusive Model Derivation (NIMD), an algorithm that automates the modeling of residential electric loads. Such models are useful for a variety of analytical techniques, such as Non-Intrusive Load Monitoring. Further, these models can be used to identify deviations from standard appliance energy profile associated with electrical faults.

Chapter 4 focussed on the difficulties in predicting the power generated in rooftop solar installations. I proposed SolarCast, a black-box approach to automatically provide site-specific solar predictions. This approach uses a Neural Network architecture to offer custom predictions that could be used for improved integration of renewable solar energy in our energy mix.

In **Chapter 5**, I proposed SolarClique, a method to detect anomalies in solar power generation. This method uses no additional inputs from multiple IoT sensors to identify a reduction in power generation. This method has implications for improving the effectiveness of monitoring solar panel infrastructure to maximize the investment in renewable sources by aiding in optimal operational performance.

Finally, **Chapter 6** looked at the challenges in identifying inefficient homes from a given population. I proposed WattHome, a novel probabilistic model to provide context to the weather-sensitivity of individual households. Further, I discussed an algorithm based on stochastic-dominance to associate a probable fault causing inefficiency in these shortlisted homes.

7.2 Future Work

In this thesis, we have demonstrated the usefulness of data-driven modeling to use IoT data from smart buildings. The opportunity to reduce our energy footprint by leveraging the increased instrumentation and monitoring infrastructure in buildings represents an exciting and important problem. Next, I will elaborate on a couple of promising future directions that can expand on the ideas presented in this thesis as networked devices become more ubiquitous in our built environments.

7.2.1 Identifying root cause of solar installation anomalies

SolarClique presented an approach to detect faults in the solar generation by comparing energy generated to colocated sites. However, there are several causes of such anomalies. Dirt, pollen, and snow block solar irradiance from falling on the panels thereby reducing generation. Physical damage causing discoloration of panels (again decreasing incident irradiance) and cracks also impacts solar production. Other factors include electrical issues such as poor sizing of accessory devices such as charge controller, inverter, etc. [93] presents a detailed study of these factors. Thus, to identify the root cause of the reduction in the solar generation, we can monitor a multitude of data points describing solar panel health such as - ambient temperature, panel temperature, wind speed, incident solar irradiance, voltage and current (along with power) from the solar panels, etc. Further, we can monitor temperature effects on the panels using thermographic imaging.

Fortunately, modern charge controllers are capable of providing real-time access to electrical properties of the solar installations through standard protocols such as ModBus [76]. Additionally, sensors such as pyranometers, panel temperature modules and visual data inputs from thermographic cameras can be leveraged to monitor the health of solar installations. Classifying faults causing a reduction in power generation through time-series data generated from such multiple sensors can be modeled as a sequence labeling problem. In machine learning, sophisticated Neural Network architectures (e.g., Long Short-term Memory) have been proposed to address these problems.

Early detection of solar generation faults has several benefits. First, it helps in increasing the lifetime of the solar installation and reducing future maintenance costs. Second, it helps to maximize the investment by aiding in optimal operational performance. Most importantly, electrical grids can become more resilient to the intermittent nature of solar energy and can allow increased renewable penetration.

7.2.2 Providing actionable feedback to customer

Data-driven models described in **WattHome** can also be used to generate reports that can help customers better understand their energy consumption patterns. Such reports can nudge users to reduce their energy consumption by providing actionable feedback (e.g., decrease setpoint temperatures in winter by 2° F to cut the energy bill by 10%). Moreover, willing consumers can provide access to energy consumption data from plug-level devices inside their homes to utilities for more customized reports yielding greater insights.

Additionally, experiments could be designed to match consumers with various incentives that encourage energy saving behavior among the most significant power consumers. The efficacy of these feedbacks can be studied using hypothesis testing. Such an exercise could help governments, policymakers, town municipalities, and utilities to design subsidies that are most effective in reducing energy consumption.

BIBLIOGRAPHY

[1] Solar Industry Data, Solar Industry Breaks 20 GW Barrier - Grows 34 http://www.seia.org/research-resources/solar-industry-data, Accessed April 2016.

[2] U.S. Energy Information Administration. (visited on May 2016). https://www.eia.gov/, 2016.

[3] When Will Rooftop Solar Be Cheaper Than the Grid? https://goo.gl/h1Ayy5, 2016. Accessed March, 2018.

[4] Dataport dataset. https://dataport.cloud/, 2017.

[5] EIA adds small-scale solar photovoltaic forecasts to its monthly Short-Term Energy Outlook. https://www.eia.gov/todayinenergy/detail.php?id=31992, 2017. Accessed March, 2018.

[6] Green button initiative. http://www.greenbuttondata.org/, 2017.

[7] Scipy Stack. http://www.scipy.org/stackspec.html, Accessed March 2018.

[8] Aftab, M., Chau, C.K., and Khonji, M. Real-time appliance identification using smart plugs: Demo abstract. In *Proceedings of the Eighth International Conference on Future Energy Systems* (2017).

[9] Ali, Mohamed Hassan, Rabhi, Abdelhamid, El Hajjaji, Ahmed, and Tina, Giuseppe M. Real time fault detection in photovoltaic systems. *Energy Procedia* (2017).

[10] Andrews, Rob W, Pollard, Andrew, and Pearce, Joshua M. The effects of snowfall on solar photovoltaic performance. *Solar Energy 92* (2013), 84–97.

[11] Apple and the Environment. http://www.apple.com/environment/renewable-energy/, Accessed November 2013.

[12] Armel, K., Gupta, A., Shrimali, G., and Albert, A. Is Disaggregation the Holy Grail of Energy Efficiency? The Case of Electricity. *Energy Policy*.

[13] ASHRAE, FUNIP. Fundamentals handbook. *IP Edition* (2013).

[14] Atsushi, Yona, and Toshihisa, Funabashi. Application of recurrent neural network to short-term-ahead generating power forecasting for photovoltaic system. In *Power Engineering Society General Meeting* (2007), Tampa, Florida, USA.

[15] Bacher, Peder, Madsen, Henrik, and Nielsen, Henrik Aalborg. Online short-term solar power forecasting. *Solar Energy* (2009).

[16] Bao, F., Liu, X., and Zhang, C. PyEEG: An Open Source Python Module for EEG/MEG Feature Extraction. *Computational Intelligence and Neuroscience 2011*.

[17] Barker, S., Kalra, S., Irwin, D., and Shenoy, P. Empirical Characterization and Modeling of Electrical Loads in Smart Homes. In *IGCC* (July 2013).

[18] Barker, S., Kalra, S., Irwin, D., and Shenoy, P. Empirical Characterization, Modeling, and Analysis of Smart Meter Data. *IEEE Journal on Selected Areas in Communications 32*, 7 (July 2014).

[19] Barker, S., Mishra, A., Irwin, D., Cecchet, E., Shenoy, P., and Albrecht, J. Smart*: An Open Data Set and Tools for Enabling Research in Sustainable Homes. In *SustKDD* (2012).

[20] Barker, S., Mishra, A., Irwin, D., Shenoy, P., and Albrecht, J. Smartcap: Flattening Peak Electricity Demand in Smart Homes. In *PerCom* (March 2012).

[21] Batra, N., Parson, O., Berges, M., Singh, A., and Rogers, A. A comparison of non-intrusive load monitoring methods for commercial and residential buildings. *arXiv preprint arXiv:1408.6595* (2014).

[22] Bellala, G., Marwah, M., Arlitt, M., Lyon, G., and Bash, C. Following the electrons: methods for power management in commercial buildings. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (2012).

[23] Canny, J. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-8*, 6 (Nov 1986).

[24] Capps, M. Energy and IOT. An Engineer's Perspective. *XRDS 22*, 2 (December 2015).

[25] Carmel, C.K., Shrimali, G., and Albert, A. Disaggregation: the holy grail of energy efficiency. *Energy Policy* (2013).

[26] Chen, D., Barker, Sean, Subbaswamy, A., Irwin, D., and Shenoy, P. Non-Intrusive Occupancy Monitoring using Smart Meters. In *BuildSys* (November 2013).

[27] Chib, S., and Greenberg, E. Understanding the Metropolis-Hastings Algorithm. *The American Statistician 49*, 4 (November 1995).

[28] Chollet, François. Keras. `https://github.com/fchollet/keras`, 2015.

[29] Chupong, C., and Plangklang, B. Forecasting Power Output of PV Grid Connected System in Thailand Without Using Solar Radiation Measurement. *Energy Procedia 9*, 0 (2011).

[30] Cleveland, Robert B, Cleveland, William S, and Terpenning, Irma. Stl: A seasonal-trend decomposition procedure based on loess. *Journal of Official Statistics* (1990).

[31] Cooper, Adam. Electric company smart meter deployments: foundation for a smart grid. *Institute for Electric Innovation. http://www. edisonfoundation. net/iei/publications/Documents/Final% 20Electric% 20Company% 20Smart% 20Meter% 20Deploym ents-% 20Foundation% 20for% 20A% 20Smart% 20Energy% 20Grid. pdf County of Sonoma. 2013a. Ordinance*, 6046 (2016).

[32] Daly, Christopher, Gibson, Wayne, Doggett, Matthew, Smith, Joseph, and Taylor, George. A probabilistic-spatial approach to the quality control of climate observations. In *Proceedings of the 14th AMS Conference on Applied Climatology, Amer. Meteorological Soc., Seattle, WA* (2004).

[33] De Wit, S. Influence of modeling uncertainties on the simulation of building thermal comfort performance. In *Building Simulation* (1997).

[34] Deline, Chris. Partially shaded operation of a grid-tied pv system. In *Photovoltaic Specialists Conference (PVSC), 2009 34th IEEE* (2009), IEEE, pp. 001268–001273.

[35] Dhimish, Mahmoud, Holmes, Violeta, and Dales, Mark. Parallel fault detection algorithm for grid-connected photovoltaic plants. *Renewable Energy* (2017).

[36] Django Project. `https://www.djangoproject.com/`, 2016.

[37] Dygraph. `http://dygraphs.com/`, 2015.

[38] Ebling, M., and Corner, M. It's all About Power and those Pesky Power Vampires. *IEEE Pervasive Computing 8*, 1 (2009).

[39] Fan, C., Xiao, F., and Wang, S. Development of prediction models for next-day building energy consumption and peak power demand using data mining techniques. *Applied Energy* (2014).

[40] Fei, H., Kim, Y., Sahu, S., Naphade, M., Mamidipalli, S.K., and Hutchinson, J. Heat pump detection from coarse grained smart meter data with positive and unlabeled learning. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (2013).

[41] Fels, M. Prism: An Introduction. *Energy and Buildings* (1986).

[42] Fontugne, R., Ortiz, J., Tremblay, N., Borgnat, P., Flandrin, P., Fukuda, K., Culler, D., and Esaki, H. Strip, bind, and search: a method for identifying abnormal energy consumption in buildings. In *Proceedings of the 12th international conference on Information processing in sensor networks* (2013).

[43] Forecast io. `https://developer.forecast.io/docs/v2`, 2016.

[44] Fu, Ran, Feldman, David J, Margolis, Robert M, Woodhouse, Michael A, and Ardani, Kristen B. US solar photovoltaic system cost benchmark: Q1 2017. Tech. rep., National Renewable Energy Laboratory (NREL), Golden, CO (United States), 2017.

[45] Gao, Peter Xiang, Golab, Lukasz, and Keshav, Srinivasan. What's Wrong with my Solar Panels: a Data-Driven Approach. In *EDBT/ICDT Workshops* (2015), pp. 86–93.

[46] Garoudja, Elyes, Harrou, Fouzi, Sun, Ying, Kara, Kamel, Chouder, Aissa, and Silvestre, Santiago. Statistical fault detection in photovoltaic systems. *Solar Energy* (2017).

[47] Gaye, Amie, et al. Access to energy and human development. *Human development report 2008* (2007).

[48] Gelman, Andrew, et al. Prior distributions for variance parameters in hierarchical models (comment on article by browne and draper). *Bayesian analysis 1*, 3 (2006), 515–534.

[49] Goiri, I., Beauchea, R., Le, K., Nguyen, T., Haque, M., Guitart, J., Torres, J., and Bianchini, R. GreenSlot: Scheduling Energy Consumption in Green Datacenters. In *SC* (November 2011).

[50] Hart, G. Nonintrusive appliance load monitoring. *Proceedings of the IEEE* (1992).

[51] Higley, L.G., Pedigo, L.P., and Ostlie, K.R. Degday: a program for calculating degree-days, and assumptions behind the degree-day approach. *Environmental entomology* (1986).

[52] Huang, R., Huang, T., Gadh, R., and Na, L. Solar Generation Prediction Using the ARMA Model in a Laboratory-level Micro-grid. `http://web.mit.edu/na_li/www//ForcastSGC2012.pdf`, 2012.

[53] Huang, Rui, Huang, Tiana, Gadh, Rajit, and Li, Na. Solar generation prediction using the arma model in a laboratory-level micro-grid. In *Smart Grid Communications (SmartGridComm), 2012 IEEE Third International Conference on* (2012), IEEE.

[54] Hygh, J.S., DeCarolis, J.F., Hill, D.B., and Ranjithan, S.R. Multivariate regression as an energy assessment tool in early building design. *Building and Environment* (2012).

[55] Insight, Berg. Smart homes and home automation. *M2M research series* (2013).

[56] Iyengar, S., Kalra, S., Ghosh, A., Irwin, D., Shenoy, P., and Marlin, B. iProgram: Inferring Smart Schedules for Dumb Thermostats. In *BuildSys* (November 2015).

[57] Iyengar, S., Lee, S., Irwin, D., and Shenoy, P. Analyzing energy usage on a city-scale using utility smart meters. In *Proceedings of the 3rd ACM International Conference on Systems for Energy-Efficient Built Environments* (2016).

[58] Iyengar, Srinivasan, Sharma, Navin, Irwin, David, Shenoy, Prashant, and Ramamritham, Krithi. A Cloud-Based Black-Box Solar Predictor for Smart Homes. *ACM Transactions on Cyber-Physical Systems 1*, 4 (2017), 21.

[59] Janetzko, H., Stoffel, F., Mittelstädt, S., and Keim, D.A. Anomaly detection for visual analytics of power consumption data. *Computers & Graphics* (2014).

[60] Katipamula, S., and Brambley, M. Review article: methods for fault detection, diagnostics, and prognostics for building systemsa review, part i.

[61] Kelso, J., Ed. *2011 Buildings Energy Data Book*. Department of Energy, March 2012.

[62] Kim, Katherine A, Seo, Gab-Su, Cho, Bo-Hyung, and Krein, Philip T. Photovoltaic hot-spot detection for solar panel substrings using ac parameter characterization. *IEEE Transactions on Power Electronics* (2016).

[63] Kissock, J., Haberl, J., and Claridge, D. Development of a toolkit for calculating linear, change-point linear and multiple-linear inverse building energy analysis models. Tech. rep., Texas A&M University, 2002.

[64] Kleiminger, W., Beckel, C., Staake, T., and Santini, S. Occupancy Detection from Electricity Consumption Data. In *BuildSys* (November 2013).

[65] Kogler, Alexander, and Traxler, Patrick. Locating faults in photovoltaic systems data. In *International Workshop on Data Analytics for Renewable Energy Integration* (2016), Springer.

[66] Kolter, J., and Ng, A. Energy Disaggregation via Discriminative Sparse Coding. In *NIPS* (December 2010).

[67] Kolter, J.Z., Batra, S., and Ng, A.Y. Energy disaggregation via discriminative sparse coding. In *Advances in Neural Information Processing Systems* (2010).

[68] Levy, H. *Stochastic dominance: Investment decision making under uncertainty*. Springer, 2015.

[69] Liu, Z., Chen, Y., Bash, C., Wierman, A., Gmach, D., Wang, Z., Marwah, M., and Hyser, C. Renewable and Cooling Aware Workload Management for Sustainable Data Centers. In *SIGMETRICS* (June 2012).

[70] Lorenz, E., Remund, J., Müller, S.C., Traunmüller, W., Steinmaurer, G., Pozo, D., Ruiz-Arias, J.A., Fanego, V.L., Ramirez, L., Romeo, M.G., Kurz, C., Pomares, L.M., and Guerrero, C. Benchmarking of Different Approaches to Forecast Solar Irradiance. In *24th European Photovoltaic Solar Energy Conference* (September 2009).

[71] Lorenz, Elke, Hurka, Johannes, Heinemann, Detlev, and Beyer, Hans Georg. Irradiance forecasting for the power prediction of grid-connected photovoltaic systems. *IEEE Journal of selected topics in applied earth observations and remote sensing* (2009).

[72] Makonin, S., Popowich, F., Bartram, L., Gill, B., and Bajic, I. AMPds: A Public Dataset for Load Disaggregation and Eco-Feedback Research. In *EPEC* (October 2013).

[73] Mandal, P., Madhira, S., haque, A. Ul, Meng, J., and Pineda, R. Forecasting Power Output of Solar Photovoltaic System Using Wavelet Transform and Artificial Intelligence Techniques. *Procedia Computer Science 12*, 0 (2012).

[74] Marcacci, S. US Solar Energy Capacity Grew An Astounding 4182010-2014. `http://cleantechnica.com/2014/04/24/us-solar-energy-capacity-grew-an-astounding-418-from-2010-2014/`, April 24th 2014.

[75] Marquez, Ricardo, and Coimbra, Carlos FM. Intra-hour DNI forecasting based on cloud tracking image analysis. *Solar Energy* (2013).

[76] Modbus, IDA. Modbus application protocol specification v1. 1a. *North Grafton, Massachusetts (www. modbus. org/specs. php)* (2004).

[77] Ndiaye, Ababacar, Kébé, Cheikh MF, Ndiaye, Pape A, Charki, Abdérafi, Kobi, Abdessamad, and Sambou, Vincent. A novel method for investigating photovoltaic module degradation. *Energy Procedia* (2013).

[78] Pearl, Judea. *Causality*. Cambridge university press, 2009.

[79] Pecan St. Inc. http://www.pecanst.org, May 2015.

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

[81] Picard, D. Testing and estimating change-points in time series. *Advances in Applied Probability 17*, 4 (1985).

[82] Pincus, S. Approximate entropy as a measure of system complexity. *Proceedings of the National Academy of Sciences 88*, 6 (1991).

[83] PVwatts. http://rredc.nrel.gov/solar/calculators/pvwatts/version1/, March 2016.

[84] Pysolar Python Library. http://pysolar.org/, 2007.

[85] PYTZ Python Library. http://pytz.sourceforge.net/, 2016.

[86] Reinhardt, A., Baumann, P., Burgstahler, D., Hollick, M., Chonov, H., Werner, M., and Steinmetz, R. On the Accuracy of Appliance Identification Based on Distributed Load Metering Data. In *SustainIT* (October 2012).

[87] Rhodes, Christopher J. The 2015 paris climate change conference: Cop21. *Science progress 99*, 1 (2016), 97–104.

[88] Sabbaghpur Arani, M, and Hejazi, MA. The comprehensive study of electrical faults in pv arrays. *Journal of Electrical and Computer Engineering 2016* (2016).

[89] Schölkopf, Bernhard, Hogg, David, Wang, Dun, Foreman-Mackey, Dan, Janzing, Dominik, Simon-Gabriel, Carl-Johann, and Peters, Jonas. Removing systematic errors for exoplanet search via latent causes. In *International Conference on Machine Learning* (2015).

[90] Schölkopf, Bernhard, Hogg, David W, Wang, Dun, Foreman-Mackey, Daniel, Janzing, Dominik, Simon-Gabriel, Carl-Johann, and Peters, Jonas. Modeling confounding by half-sibling regression. *Proceedings of the National Academy of Sciences* (2016).

[91] Seem, J.E. Using intelligent data analysis to detect abnormal energy consumption in buildings. *Energy and buildings* (2007).

[92] Shao, X., and Zhang, X. Testing for change points in time series. *Journal of the American Statistical Association 105*, 491 (2010).

[93] Shapiro, Daniel, Robbins, C., and Ross, P. Solar pv operation & maintenance issues, 2014.

[94] Sharma, N., Sharma, P., Irwin, D., and Shenoy, P. Predicting Solar Generation from Weather Forecasts Using Machine Learning. In *SmartGridComm* (October 2011).

[95] Sproul, A. Derivation of the solar geometric relationships using vector analysis. *Renewable Energy 32*, 7 (2007).

[96] Stettler, S, Toggweiler, P, Wiemken, E, Heydenreich, W, de Keizer, AC, van Sark, WGJHM, Feige, S, Schneider, M, Heilscher, G, Lorenz, E, et al. Failure detection routine for grid-connected pv systems as part of the pvsat-2 project. In *Proceedings of the 20th European Photovoltaic Solar Energy Conference & Exhibition, Barcelona, Spain* (2005), pp. 2490–2493.

[97] Tahri, Ali, Oozeki, Takashi, and Draou, Azzedine. Monitoring and evaluation of photovoltaic system. *Energy Procedia* (2013), 456–464.

[98] Tao, C., Shanxu, D., and Changsong, C. Forecasting Power Output for Grid-connected Photovoltaic Power System Without Using Solar Radiation Measurement. In *PEDG* (June 2010).

[99] Traxler, Patrick. Fault detection of large amounts of photovoltaic systems. In *Proceedings of the ECML/PKDD 2013 Workshop on Data Analytics for Renewable Energy Integration* (2013).

[100] Wang, S., Yan, C., and Xiao, F. Quantitative energy performance assessment methods for existing buildings. *Energy and Buildings* (2012).

[101] Woyte, Achim, Richter, Mauricio, Moser, David, Mau, Stefan, Reich, Nils, and Jahn, Ulrike. Monitoring of photovoltaic systems: good practices and systematic analysis. In *Proc. 28th European Photovoltaic Solar Energy Conference* (2013).

[102] Weather underground. `https://www.wunderground.com`, March 2016.

[103] Yan, C., Wang, S., and Xiao, F. A simplified energy performance assessment method for existing buildings based on energy bill disaggregation. *Energy and buildings* (2012).

[104] Yona, A., Senjyu, T., and Funabashi, T. Application of Recurrent Neural Network to Short-term-ahead Generating Power Forecasting for Photovoltaic System. In *IEEE Power Engineering Society General Meeting* (June 2007).

[105] Zehner, O. Unclean at Any Speed. *IEEE Spectrum 50* (July 2013).

[106] Zhao, H., and Magoulès, F. A review on the prediction of building energy consumption. *Renewable and Sustainable Energy Reviews* (2012).

[107] Zhou, Q., Wang, S., and Ma, Z. A model-based fault detection and diagnosis strategy for hvac systems. *International Journal of Energy Research* (2009).