A Cloud-based Black Box Solar Predictor for Smart Homes[1]

SRINIVASAN IYENGAR, University of Massachusetts Amherst
NAVIN SHARMA, University of Massachusetts Amherst
DAVID IRWIN, University of Massachusetts Amherst
PRASHANT SHENOY, University of Massachusetts Amherst
KRITHI RAMAMRITHAM, Indian Institute of Technology Bombay, India

The popularity of rooftop solar for homes is rapidly growing. However, accurately forecasting solar generation is critical to fully exploiting the benefits of locally-generated solar energy. In this paper, we present two machine learning techniques to predict solar power from publicly-available weather forecasts. We use these techniques to develop SolarCast, a cloud-based web service, 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.

We evaluate the accuracy of SolarCast's different algorithms on two publicly available datasets, each containing over one hundred rooftop deployments with a variety of attributes, e.g., climate, tilt, orientation, etc. We show that SolarCast learns a more accurate model using much less data ($\sim$1 month) than prior SVM-based approaches, which require $\sim$3 months of data. SolarCast also provides a programmatic API, enabling developers to integrate its predictions into energy-efficiency applications. Finally, we present two case studies of using SolarCast to demonstrate how real-world applications can leverage its predictions. We first evaluate a "sunny" load scheduler, which schedules a dryer's energy usage to maximally align with a home's solar generation. We then evaluate a smart solar-powered charging station, which can optimally charge the maximum number of electric vehicles (EVs) on a given day. Our results indicate that a representative home is capable of reducing its grid demand up to 40% by providing a modest amount of flexibility (of $\sim$5 hours) in the dryer's start time with opportunistic load scheduling. Further, our charging station uses SolarCast to provide EV owners the amount of energy they can expect to receive from solar energy sources.

CCS Concepts: •**Information systems** → *Data analytics;* •**Computing methodologies** → *Supervised learning by regression;* Neural networks; •**General and reference** → Empirical studies;

Additional Key Words and Phrases: Solar Power Prediction, Renewable Integration, Smart Grid, Machine Learning

ACM Reference Format:
ACM Trans. Cyber-Phys. Syst. 000, 000, Article 00 (2016), 25 pages.
DOI: http://dx.doi.org/10.1145/3004056

1. INTRODUCTION

Solar generation capacity is experiencing a dramatic increase worldwide, with its peak capacity having risen five-fold over the past five years from $\sim$2300 megawatts (MW) in 2010 to $\sim$12000MW in 2014 in the U.S alone [Marcacci 2014]. Over 4750MW (or 49%) of this new capacity has been installed in just the last year alone, due, in part, to a rapid drop in solar panel prices, which have fallen 60% since 2011 [fac 2014].

[1]A conference version of this work appeared at the ACM Buildsys 2014 conference [Iyengar et al. 2014].

DOI: http://dx.doi.org/10.1145/3004056

Solar deployments come in a wide variety of sizes, ranging from the massive solar farms deployed by utilities (and some datacenter operators [Apple 2013]) to small and medium-sized deployments deployed by homeowners, farmers, and local businesses. Overall, nearly half of aggregate solar capacity now derives from small-scale home deployments ($<$10kW), many of which rely on *net metering* to transfer surplus energy to the grid [Marcacci 2014], 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. Solar energy forecasts have a wide range of applications. For example, smart buildings could employ forecasts for opportunistic scheduling of elastic loads, while utilities could use them to estimate aggregate demand across a customer base with a high penetration of solar energy.

In this paper, we argue that 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, which 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 in order 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 future solar output.

Unfortunately, the characteristics 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 actually more complex and dynamic than for large solar farms, they require even more time and expertise to develop. In addition, 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 [sol 2016].

Despite the challenges above, accurate near-term predictions, if available, have the potential to yield numerous benefits. Effective planning is key to maximizing the environmental benefits of solar energy and enabling it to scale to a significant fraction of homes. 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.

As a result, most states place strict limits on both i) the amount of power individual consumers may net meter and ii) the total fraction of generation contributed by net metering [Barnes et al. 2013]. In addition, there is also a concern that net metering may actually increase the electricity costs of consumers without solar installations [Petlin and Wu 2013], since the increased generation costs from satisfying a more stochastic demand are distributed across all consumers [Kaplan 2008].

Since solar energy today only contributes a small fraction of grid energy ($\sim$1% [Marcacci 2014]), the lack of effective planning does not currently pose a significant problem. However, as solar penetration rises, effective planning by homes and utilities will be increasingly important to maintain grid stability. To facilitate better planning at large scales, we present SolarCast, an open cloud service that accurately forecasts near-term energy generation for the increasing number of small-scale residential solar deployments. Unlike prior work on solar forecasting, which often relies on detailed site-specific information to develop forecast models, SolarCast *automatically* generates prediction models using a "black box" approach that only requires i) a site's location and ii) a *minimal* amount of historical generation data. Our black box approach follows from SolarCast's design as a cloud-based service: since most homeowners do not know the arcane details of their solar installation required by many models, such as the efficiency and type of their panels or their orientation, we learn these parameters when possible, and dynamically adjust predictions to account for their impact, when not. Since the numerous new solar installations that are rapidly coming online have little historical data, SolarCast also focuses on reducing the amount of historical data necessary for accurate modeling, which has not been explicitly addressed in prior work. Instead, prior work often assumes historical generation data is available for each time of the year, which implicitly requires *multiple years* of data, since the maximum solar generation capacity varies at each time of the day on each day of the year. In addition, SolarCast is a live service that continuously updates both its i) model based on fresh data and ii) predictions based on new weather forecasts. In designing SolarCast, we make the following contributions.

Automatic Model Generation. We develop a system for *automatically* generating accurate prediction models customized for each solar deployment, while also minimizing the amount of i) information about the deployment and i) historical generation data necessary to generate such models. Our system leverages these techniques and uses continuous learning to refine the model as new data becomes available.

Black Box Prediction Model. We employ "black box" model for each solar site that learns important static and dynamic parameters of the installation. Static parameters include the panels' tilt and orientation, while dynamic parameters include dust and pollen, snow cover, and shade from leaves or buildings. We present and compare results from building prediction models using two machine learning techniques: i) a custom non-linear least square curve fit, and ii) a deep neural network model. These techniques automatically learn the static site-specific parameters of a deployment, e.g., tilt and orientation, while adjusting for errors online due to dynamic parameters.

To minimize the need for historical generation data, we normalize each point in time relative to the ideal cloudless irradiance based on the location and angle of the sun. Our key insight is that the same weather parameters affect ideal cloudless irradiance by the same proportion at any time, i.e., the same weather will reduce the ideal irradiance by 50% at both 5pm and 12pm. This normalization enables SolarCast to learn a single model for any granularity of data, rather than multiple models for different periods of time. By better modeling of the domain, we are able to prepare a single model

across different time periods and thus substantially increase the amount of training data available for predicting solar power at any time period. In contrast, prior work builds a separate model for each time of the day and day of the year [Sharma et al. 2011]. This insight also enables our technique to be agnostic to data resolution. As we show, our models can predict solar power at high data resolutions, e.g. every minute for the next hour, give accurate weather forecasts at those resolutions.

Scalable Cloud Service. We design SolarCast as an open cloud service that provides customized predictions for each solar site based on models learned from both historical data and continuously refined using live data. In addition, the service also offers a web-based API to enable application developers to incorporate SolarCast predictions into their energy management applications.

Implementation and Evaluation. We evaluate SolarCast's two prediction models on two publicly available datasets, each containing over one hundred rooftop deployments varying in the installation size from different locations (and climates) across the U.S. In addition, we also evaluate the performance of our models on three utility-scale solar farms. Our results show that SolarCast learns a more accurate model significantly faster (using less training data) than an approach based strictly on Support Vector Machines (SVMs), which requires a different model for each time period. In addition, we present two case studies. Our first case study implements sunny load scheduling to maximally align a dryer's operation (typically the largest load in a home) with solar generation. Our second case study examines a solar-powered EV charging station, which provides EV owners an estimate of the amount of charging they can expect to receive from green solar energy. In the first case study, we find that a representative home is capable of reducing its grid energy demand up to 40% by providing a modest amount of flexibility (of $\sim$5 hours) to defer the dryer's start time. In the second case study, we show that SolarCast provides EV owners an accurate measure of solar charging capacity.

2. BACKGROUND AND MOTIVATION

Our work focuses primarily on rooftop solar deployments for smart homes and buildings, although we note that our techniques are applicable to other small-to-medium-scale solar deployments, e.g., such as ground-based solar arrays. We assume that the smart home or building uses its locally generated solar energy whenever possible and net meters any excess solar energy to the grid. The goal of our work is to build a system that provides short-term forecasts of solar generation for the next few hours or days that are customized to each site. Other applications may use these predictions to optimize a building's energy usage. To provide customized forecasts, our system automatically learns a custom prediction model for each solar installation that accounts for many of the factors that influence solar generation. The primary factor that governs solar output is the solar irradiance, or the amount of sunlight visible by the panels.

As is well known, solar generation is intermittent since solar irradiance itself depends on many factors, including the weather, e.g., a sunny versus cloudy day. Solar generation also depends on other factors, as shown in Table 1, including: i) *panel characteristics*, such as their size, type, age, and number, ii) *site characteristics*, such as panel placement, tilt, and orientation, as well as the geographic location of the installation, iii) *surrounding characteristics*, such as nearby trees, neighboring buildings or other structures that may cast shadows on panels; the amount of shadows will themselves vary by the time of the day, as well as by the seasons and foliage, iv) *seasonal characteristics*, such as the season of the year, which determines the length of the day and the solar intensity, v) *weather characteristics* which encompass a variety of weather parameters including cloud cover and temperature, and vi) *other dynamic characteristics* such as dust or snow on the panels, which vary over time. In addition,

electrical characteristics also affect generation for net metered deployments, as solar output depends on the inverter's operating voltage and the grid's voltage. While we discuss these factors in more detail, we note that many of them are highly specific to a particular installation. Thus, precisely quantifying them is not practical, or even possible, for owners of solar deployments having limited technical background.

Thus, for SolarCast, we assume knowledge of only the GPS coordinates of the installation, specified in the form of a street address, and a minimal amount of training data, which records recent history of solar generation at a site. We note that, unlike some prior efforts that require a substantial amount of training data to learn a model, our goal is to minimize the amount of training data needed to learn an initial model. We impose this requirement since an explicit goal of our system is to provide accurate predictions even to *new* rooftop solar installations that may have only a small amount of historical data. Since new solar installations are coming online rapidly, many do not have significant historical data. Since many solar installations provide live generation data through web-based interfaces, we assume that our system may also use live data, when available, to continually refine its models and adjust its predictions.

A final design goal is automation and scale. Since there are a large number of small solar deployments, the model generation process must be automated to scale to many deployments and provide continuous predictions. Since each site has unique characteristics, each generated model will be different, which requires our system to learn a unique model for each site. Lastly, we note that the generated models are themselves likely to be more complex than those for large solar farms which have "better optimized" installations that may permit static models. In contrast, a static model may not suffice for rooftop installations when accounting for the impact of dynamic parameters, such as shade, foliage, dust, and snow. Below we detail how weather conditions and configuration parameters affect solar power output, as well as our motivation for considering a black box approach for predicting solar generation.

2.1. Impact of Weather Conditions

The power generated by a solar panel depends on the amount of light, i.e., solar radiation, incident on its surface. The radiation from the sun passes through the earth's atmosphere, resulting in some loss of energy due to scattering by atmospheric components in the sky, such as cloud cover, vapor, dust, and pollen. The incident radiation, called *insolation*, has a direct bearing on the amount of solar power generated. Several other weather parameters, such as humidity, dew point, temperature, and precipitation, also affect insolation. Sharma et al. [Sharma et al. 2011] discuss how different weather parameters affect solar generation and quantify their correlation with solar irradiance. In addition, weather conditions also affect the panel properties. As with any semiconductor device, solar cells are sensitive to temperature. As the temperature increases, the voltage across each cell reduces. The cell temperature tends to be higher than the ambient temperature by $\sim$20°C. Also, every $1^{\circ}C$ rise in the cell temperature above $25^{\circ}C$ reduces the efficiency by 0.5% [Skoplaki and Palyvos 2009]. Further, humidity causes ingression to the solar cell enclosure. Mekhilef et al. [Mekhilef et al. 2012] detail the impact of humidity, air velocity, and dust on the panel's efficiency.

2.2. Impact of Configuration Parameters

Apart from atmospheric conditions, installation parameters, such as tilt and orientation of the panels, also affect the power generation. If the tilt and orientation of a panel is known, a simple trigonometric formula exists, as discussed in the next section, to derive insolation incident on the panel for the given tilt, orientation, and location of the installation. Unfortunately, while large solar farms may know the precise tilt and orientation of their panels, owners of rooftop installations may not know the precise

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

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

value of the tilt and orientation. In addition, rooftop tilt and orientation is rarely ideal, and is often dictated by site-specific issues, such as avoiding nearby tree shadows and optimizing roof space. As a result, these parameters tend to vary widely for different sites. Further, other site specific factors, such as the inverter efficiency, battery capacity and condition, and grid connectivity, also affect the power generated from a solar panel. Technical manuals for photovoltaic (PV) systems [inverter 2014] discuss these issues in detail. As with tilt and orientation, a typical homeowner is also not likely to know these more detailed parameters for a typical rooftop installation.

2.3. Our Black Box Approach

Prior work [Sharma et al. 2011] has focused on using machine learning techniques to automatically learn the impact of weather conditions, while assuming that panel properties and configuration parameters are known in advance. However, for residential rooftop installations, it is often difficult to obtain the panel properties and installation-specific configuration parameters because homeowners generally 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 which panel properties are static and which are dynamic.

To generate solar power prediction models at scale, for any solar installation in the country, we 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. Our black box model automatically learns static parameters, such as tilt and orientation, using a grid search technique, and then uses a machine learning technique to predict power from weather forecasts. In addition, our black box model also employs an adaptive learning algorithm to account for dynamic properties, and to continue refining the model as and when new data becomes available.

3. 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 prior work on machine learning-based solar predictions, SolarCast also employs machine learning techniques to derive its model. However, the primary dif-

ference 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. In this case, 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 learned model is then a "function" that, given certain inputs, such as weather and time of day, will compute the expected solar output under those conditions. Much of the prior work, including some of our own [Sharma et al. 2011], takes such an approach. In contrast, in a black box approach, the solar deployment is assumed to 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 in advance are accounted for by adjusting the predictions dynamically based on live data. 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).

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:

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

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.

$$MAPE = \frac{100}{n} \cdot \sum_{t=1}^{n} \left| \frac{A_t - P_t}{A^{'}} \right| \tag{2}$$

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.

3.1. Model Generation

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.

3.1.1. 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 [Sharma et al. 2011], 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 [Sharma et al. 2011], 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 mul-

tiplying 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:

$$P_t = f\Big(S_t^{cloudless} \cdot W_t\Big) \tag{3}$$

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 [Sharma et al. 2011]. 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.

3.1.2. 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 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:

$$P_t = f'\Big(S_t^{module} \cdot \nu_t \cdot W_t\Big) \tag{4}$$

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:

$$S_t^{module} = S_t^{cloudless} \cdot (\cos\alpha_t \sin\beta \cos(\psi - \theta_t) + \sin\alpha_t \cos\beta) \tag{5}$$

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 1). The details involved in deriving this equation can be found in [Sproul 2007]. 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

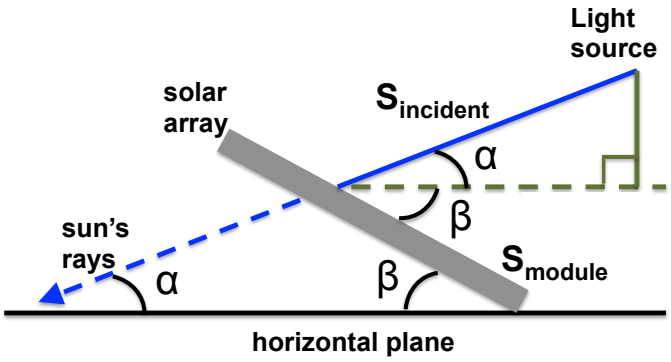


Fig. 1. Tilt of panel (β) and solar elevation (α)

different adjustment factors discussed in the literature, we have the following general equation to account for temperature changes with hyperparameters a and b:

$$\nu_t = a \cdot (T_t^{cell} - b) \tag{6}$$

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.

3.2. 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 historical 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.

3.2.1. 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.

3.2.2. Model variation. The function f, described in Equation 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 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:

$$P_t = f'\left(S_t^{module} \cdot \nu_t \cdot W_t, P_{t-24h}^{output}\right) \tag{7}$$

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 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.

3.3. 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 2 to learn the function defined by Equation 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

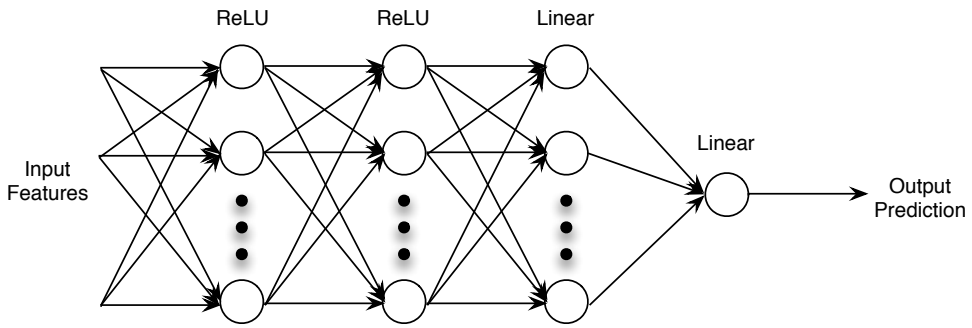


Fig. 2. SolarCast's Deep Neural Network Architecture

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)*.

3.3.1. Hyperparameter Learning. We use a grid search technique to automatically find the tilt and orientation 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 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.

3.4. 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, the online model rapidly converges, within 10%, to the static model within one month. This illustrates

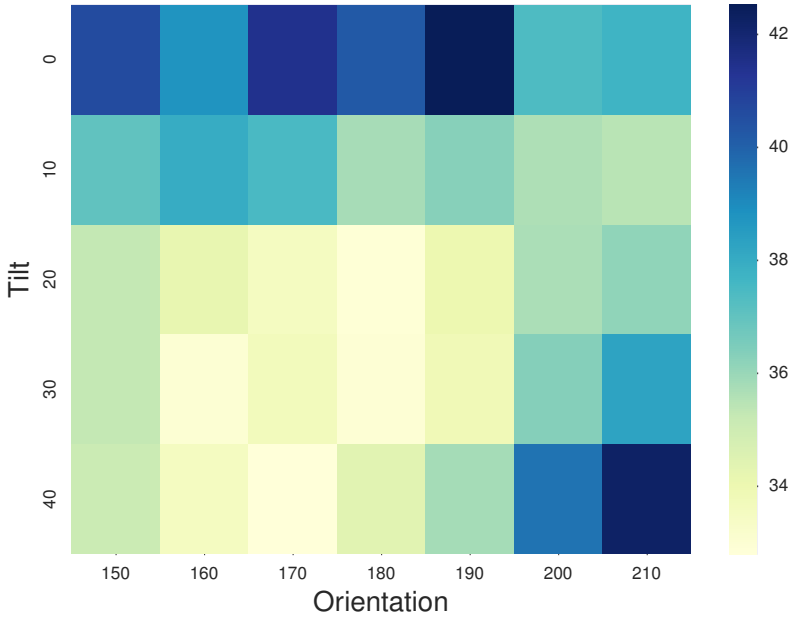


Fig. 3. Grid Search over different values of Tilt and Orientation

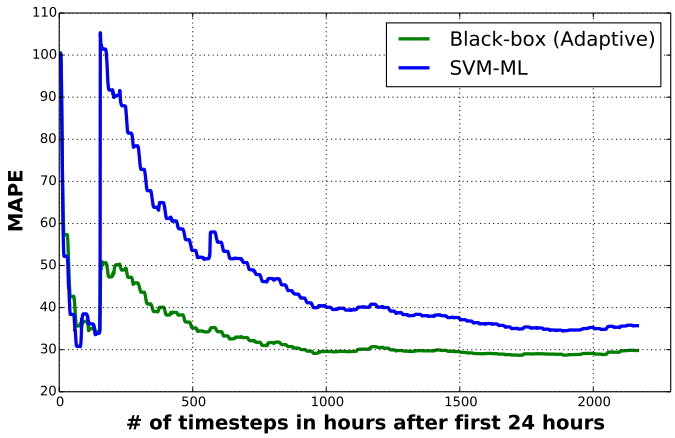


Fig. 4. Prediction error for the online version of SolarCast's adaptive model and SVM-ML model.

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 [Sharma et al. 2011] 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

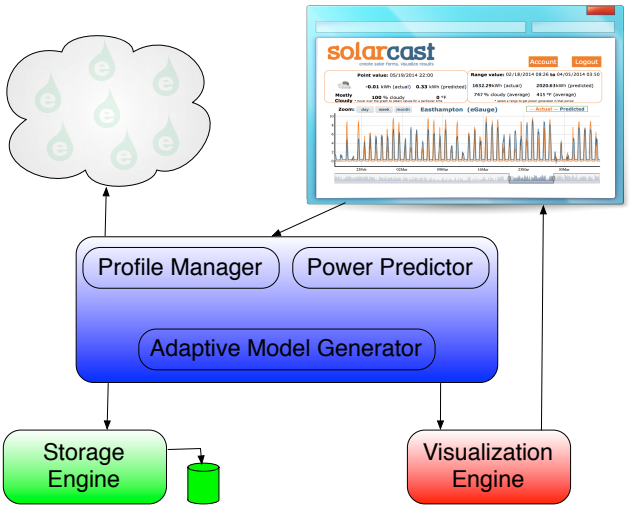


Fig. 5. SolarCast Web Service Architecture

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. SOLARCAST CLOUD SERVICE

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

4.1. 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 [pvwatts 2016], which are either proprietary or require knowing installation- and panel-specific parameters, 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.

4.2. Implementation

We use many open source libraries to build SolarCast and its black box prediction model. We use Django [django 2016], an open source web application framework written in Python, to build SolarCast's web service. The visualization engine uses dygraph [dygraph 2015], 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 [Chollet 2015], an open source deep learning library in python. In addition, we use libraries – SciPy, NumPy, Pandas – from the SciPy stack [scipy 2016] 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 [pytz 2016], which automatically handles the daylight saving issues. To get weather forecasts for any location we use the Forecast API from Forecast.io[forecast 2016]. 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,

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

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

and the cloudless irradiance at a particular time for a given location. We use the PySolar library [pysolar 2007] that implements the Masters' algorithm.

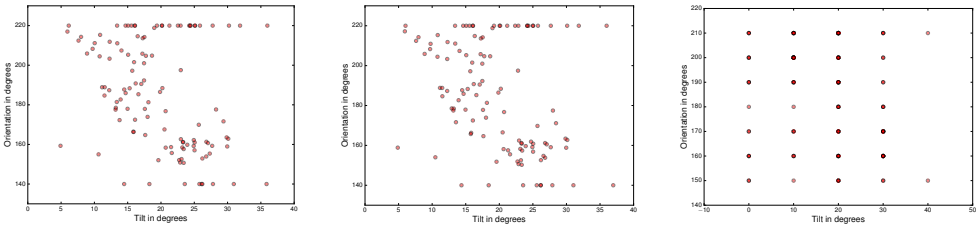
5. EVALUATION

We evaluate our black box prediction model on three geographically diverse datasets. Table 5 describes them in terms of their number of installations, duration, data granularity and installed capacity. All the installations in the Pecan Street dataset [pecan 2015] are located in 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 [Barker et al. 2012]). 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. For these installations, accounts were created on the SolarCast web service to visually validate our predictions.

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) [Sharma et al. 2011]. 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 with the linear kernel. As opposed to predicting irradiance, we directly predict power based on weather forecasts for each day.

5.1. 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.



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

Fig. 6. Tilt and orientation for different solar installation in the Pecan Street dataset.

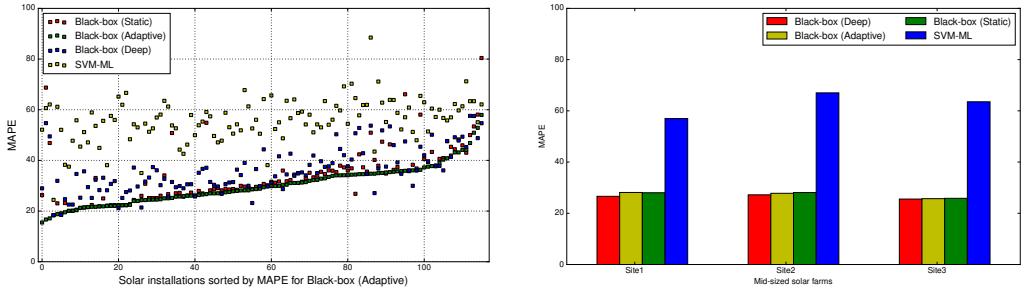
Figure 6 shows the tilt and orientation for the three different algorithms for each site in the Pecan Street dataset, where the x-axis is the tilt and the y-axis is the orientation. As the figures show, the tilt and orientation vary greatly across different sites, which highlights the importance of an automatic technique like our black box model to learn the configuration parameters rather than assuming fixed values for all sites.

5.2. Model Comparison

5.2.1. 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 [Sharma et al. 2011]. We use MAPE to measure the prediction accuracy of each model. Figure 7(a) plots MAPE for all four models for 116 rooftop installations for the third-party dataset, while Figure 7(b) plots the same for the three open-space medium-sized utility solar farms, each with over .8MW capacity. In Figure 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$3-5%) but overall seems to produce inferior results. In Figure 7(b), we find that the neural network approach performs the best for all three locations by ($\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.

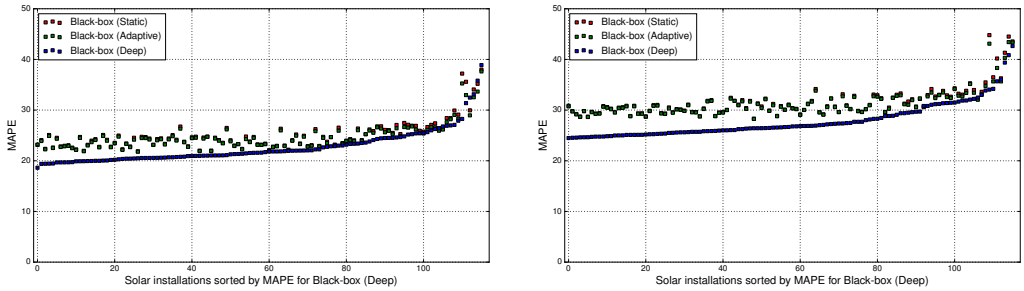
Figure 8(a) shows the comparison between our three black box approaches on the Pecan Street dataset with hour-level forecasts. For almost half the sites the deep neural network based approach outperforms the static and the adaptive approach by ($\sim$3-5%). In most other cases, the performance is ($\sim$1-3%) better.

5.2.2. 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 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 fore-



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

Fig. 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

Fig. 8. Prediction error for various prediction models for each site in Pecan Street dataset over 1 years.

cast. 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 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%).

5.3. Case Studies

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.

5.3.1. 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

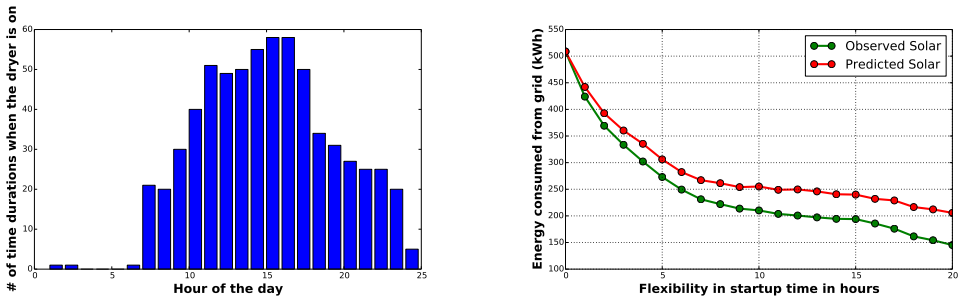


Fig. 9. Frequency distribution for hour of day when the dryer is run (a) and grid demand for start time flexibility (b).

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. Figure 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 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.

Result: *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 ∼5 hours to a dryer's startup time.*

5.3.2. 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 [Zehner 2013], 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 paper, 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.

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

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

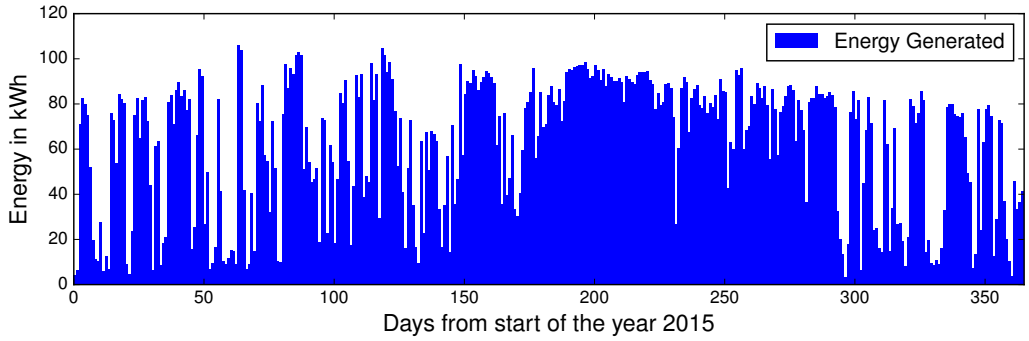


Fig. 10. Solar energy produced by the smart charging station over the year 2015

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 [pecan 2015]. 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 to the amount of demand shown in the data (see Figure 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 10 shows the energy produced by the solar charging station over the period of a year. Table 5.3.2 describes the maximum charging rate and the energy consumed over the period of a year by the EVs. Figure 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 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 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

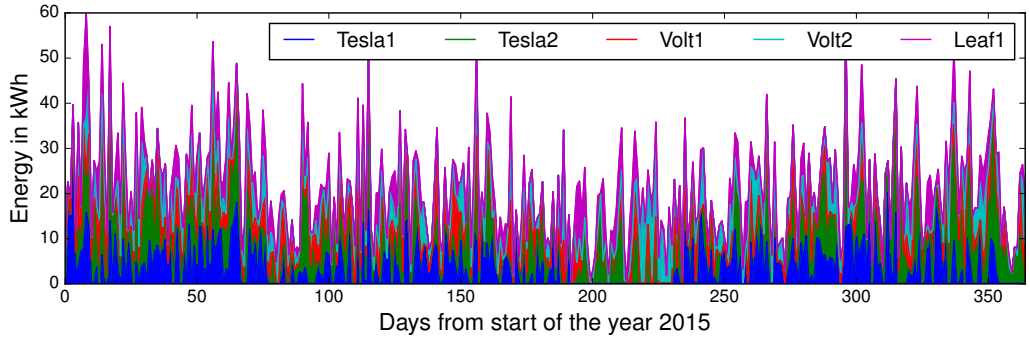


Fig. 11. Energy demand profile of the different EVs in our study over the period of the year 2015

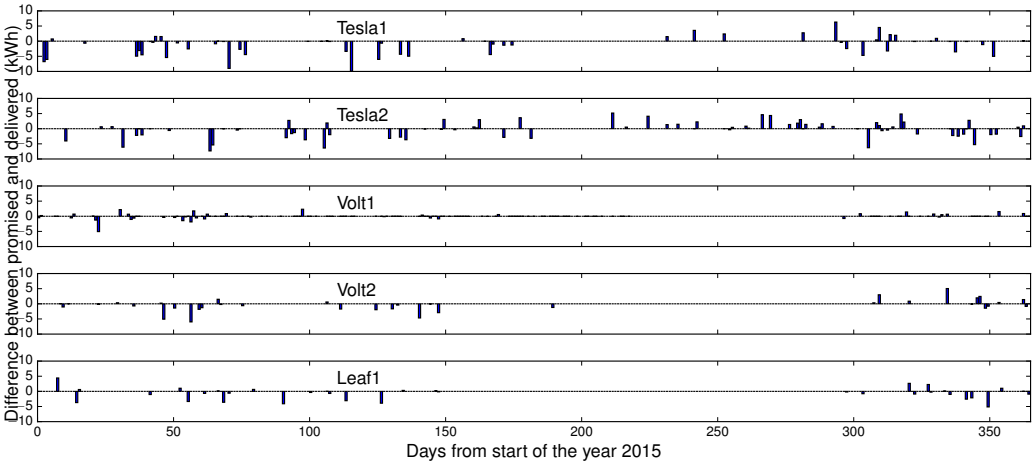


Fig. 12. Mismatch between the promised and the delivered energy for the different EVs over the whole year

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.

Result: *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*

6. LIMITATIONS OF THE MODEL AND FUTURE DIRECTIONS

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 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 [wunder 2016] weather station.

Clear Sky days: Figure 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%.

Overcast days: Figure 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.

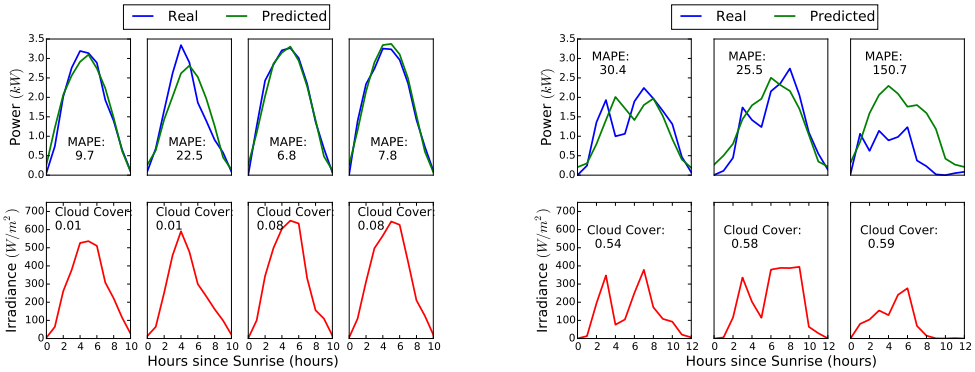
Observations:

(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).

Future Directions: In this work, we focussed on rooftop installations. Unlike these installations, utility scale installations leverage additional ground sensors such as sky imagers or photo sensors [Achleitner et al. 2014] to predict future power generation. With these systems becoming more ubiquitous, we can leverage these to make better predictions. Further, as mentioned earlier, cloud cover values are inadequate to model solar power. However, with increasing number of on ground sensors, we would benefit from more detailed weather forecasts such as the direction of clouds (to know if this will block the sun or not), the thickness of clouds etc.

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. 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. [Lorenz et al. 2009] and Huang et al. [Huang et al. 2012] 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. [Yona et al. 2007] use a neural network model to forecast solar irradiance; they then use a site-specific irradiance → power model to forecast power generation.



(a) Sunny days (b) Cloudy days

Fig. 13. Error analysis for the two sets of days

Tao et al. [Tao et al. 2010] 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 [Chupong and Plangklang 2011] that uses Elman Neural Networks with an input layer very similar to that of related work [Tao et al. 2010]. Mandal et al. [Mandal et al. 2012] presents a hybrid model that uses wavelets and Neural Networks. Moreover, all of these techniques have used a single site and limited dataset (∼4 days) for evaluation. Apart from neural network models, machine learning (ML) based statistical techniques [Sharma et al. 2011; Goiri et al. 2011; Liu et al. 2012] 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.

8. CONCLUSION

In this paper, 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 elec-

tricity 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.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable feedback and the editors for their constructive inputs. This research was supported in part by NSF grants IIP-1534080, CNS-1405826, CNS-1253063, CNS-1505422, and the Massachusetts Department of Energy Resources. It is also supported in part by a grant from Tata Consultancy Services to IIT Bombay. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

REFERENCES

2014. Solar Energy Industries Association, Solar Energy Facts: 2013 Year In Review. http://www.seia.org/sites/default/files/YIR%202013%20SMI%20Fact%20Sheet.pdf. (March 5th, 2014 2014).

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

S. Achleitner, A. Kamthe, T. Liu, and A. Cerpa. 2014. SIPS: Solar Irradiance Prediction System. In *IPSN*.

Apple 2013. Apple and the Environment. http://www.apple.com/environment/renewable-energy/. (Accessed November 2013).

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

J. Barnes, T. Culley, R. Haynes, L. Passera, J. Wiedman, and R. Jackson. 2013. *Best Practices in State Net Metering Policies and Interconnection Procedures*. Technical Report. Freeing the Grid.

François Chollet. 2015. Keras. https://github.com/fchollet/keras. (2015).

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

django 2016. Django Project. https://www.djangoproject.com/. (2016).

dygraph 2015. Dygraph. http://dygraphs.com/. (2015).

forecast 2016. Forecast IO. https://developer.forecast.io/docs/v2. (2016).

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

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

inverter 2014. Inverter, Storage and PV System Technology Industry Guide 2014. http://www.pv-system-tech.com/. (2014).

S. Iyengar, N. Sharma, D. Irwin, P. Shenoy, and K. Ramamkritham. 2014. SolarCast - A Cloud-based Black Box Solar Predictor for Smart Homes. In *BuildSys*.

Stan Kaplan. 2008. *Power Plants: Characteristics and Costs*. Technical Report. Congressional Research Service Report to Congress.

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

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

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

S. Marcacci. 2014. 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).

S. Mekhilef, R. Saidur, and M. Kamalisarvestani. 2012. Effect of Dust, Humidity and Air Velocity on Efficiency of Photovoltaic Cells. *Renewable and Sustainable Energy Reviews* 16, 5 (2012).

pecan 2015. Pecan St. Inc. http://www.pecanst.org. (May 2015).

G. Petlin and K. Wu. 2013. *California Net Energy Metering Ratepayer Impacts Evaluation*. Technical Report. California Public Utilities Commission.

pvwatts 2016. PVwatts. http://rredc.nrel.gov/solar/calculators/pvwatts/version1/. (March 2016).

pysolar 2007. PySolar Python Library. http://pysolar.org/. (2007).

pytz 2016. PYTZ Python Library. http://pytz.sourceforge.net/. (2016).

scipy 2016. SciPy Stack. http://www.scipy.org/stackspec.html. (2016).

N. Sharma, P. Sharma, D. Irwin, and P. Shenoy. 2011. Predicting Solar Generation from Weather Forecasts Using Machine Learning. In *SmartGridComm*.

E. Skoplaki and J.A. Palyvos. 2009. On the Temperature Dependence of Photovoltaic Module Electrical Performance: A Review of Efficiency/Power Correlations. *Solar Energy* 83, 5 (2009).

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

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

wunder 2016. Weather Underground. https://www.wunderground.com. (March 2016).

A. Yona, T. Senjyu, and T. Funabashi. 2007. Application of Recurrent Neural Network to Short-Term-Ahead Generating Power Forecasting for Photovoltaic System. In *IEEE Power Engineering Society General Meeting*.

O. Zehner. 2013. Unclean at Any Speed. *IEEE Spectrum* 50 (July 2013).