SolarCast - A Cloud-based Black Box Solar Predictor for Smart Homes

Srinivasan Iyengar, Navin Sharma, David Irwin, Prashant Shenoy
University of Massachusetts Amherst

Krithi Ramamritham
Indian Institute of Technology Bombay, India

Abstract

The popularity of rooftop solar for individual homes continues to rise rapidly. However, techniques for accurately forecasting solar generation are critical to fully exploiting the benefits of such locally-generated solar energy. In this paper, we present SolarCast, a cloud-based web service, which automatically generates models that provide customized site-specific predictions of future 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 SolarCast's accuracy on a dataset consisting of 118 geographically-diverse solar deployments, and show that it learns an accurate model using much less data ($\sim$1 month) than a prior SVM-based approach, which requires $\sim$3 months of data. SolarCast also provides a programmatic API, enabling developers to integrate its predictions directly into energy-efficiency applications. We present a case study of using SolarCast to implement one such application: a "sunny" load scheduler, which schedules a dryer's energy usage to maximally align with a home's solar generation. 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.

Categories and Subject Descriptors

J.7 [**Computer Applications**]: Computers in Other Systems—*Consumer products*

General Terms

Design, Experimentation, Measurement

Keywords

Energy, Electricity, Grid

BuildSys'14, November 5–6, 2014, Memphis, TN, USA.

1 Introduction

Solar generation capacity is experiencing a dramatic increase worldwide, having risen five-fold over the past five years from $\sim$2300 megawatts (MW) in 2010 to $\sim$12000MW in 2014 in the U.S alone [16]. Over 4750MW (or 49%) of this new capacity has been installed in just the last year alone, due, in part, to a rapid drop in solar panel prices, which have fallen 60% since 2011 [1]. Solar deployments come in a wide variety of sizes, ranging from the massive solar farms deployed by utilities (and some datacenter operators [2]) to small and medium-sized deployments by homeowners, farmers, and local businesses. Overall, nearly half of aggregate solar capacity now derives from small-scale home deployments ($<$10kW), many of which rely on *net metering* to transfer surplus energy to the grid [16], 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 hours to days are 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 constrains 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 genera-

tion 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.

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 decrease the surplus energy they net meter to the grid. 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 home 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 [4]. In addition, there is also a concern that net metering may actually increase the electricity costs of consumers without solar installations [18], since the increased generation costs from satisfying a more stochastic demand are distributed across all consumers [12].

Since solar energy today only contributes a small fraction of grid energy ($\sim$1% [16]), 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 home 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 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 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 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 both information about the deployment and historical generation data necessary to generate such models. Our system uses continuous learning to refine the model as new data becomes available.

Black Box Prediction Model. We employ "black box" models for each solar site that learns important static and dynamic parameters of the installation. Static parameters include the panels' orientation, while dynamic parameters include dust and pollen, snow cover, and shade from leaves or buildings. Our model employs grid search to learn the static site-specific parameters of a deployment, e.g., tilt and azimuth, 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, rather than multiple models for different periods of time.

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 prediction model on a geographically distributed dataset of 118 solar installations with associated location information. Our results show that SolarCast's model 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 a case study of using SolarCast to implement one such application: a "sunny" load scheduler, which schedules a dryer's energy usage to maximally align with a home's solar generation. Our results indicate 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.

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 the 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 encompasses 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. While we discuss these factors in more detail below, 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 our system, we assume knowledge of only the GPS coordination 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 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 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 a solar panel generates derives from 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. In prior work, Sharma et al.[23] 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 around $20^{\circ}C$. Also, every $1^{\circ}C$ rise in the cell temperature above $25^{\circ}C$ reduces the efficiency by 0.5% [24]. Further, humidity causes ingression to the solar cell enclosure. Mekhilef et al. [17] 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 formulae 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 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 [11] 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 [23] has focused on using machine learning techniques to automatically learn the impact of weather con-

Parameters	*Variability*	*Our Mechanism*
Panel Parameters	Static	ML Regression
Tilt	Static	Grid Search
Orientation	Static	Grid Search
NOCT	Static	Grid Search
Tree Shade	Dynamic	Adaptive Learning
Snow/Dust/Pollen	Dynamic	Adaptive Learning

Table 1. SolarCast employs different techniques to capture variations in panel and configuration parameters.

ditions, 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.

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 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 aspects. 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 learn 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 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. 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 the correlations 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 [23], 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).

We first use a machine learning regression technique to design 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. Next, we extend the model to automatically learn the configuration (and static) parameters, such as tilt, orientation, and Nominal Operating Cell Temperature (NOCT), for any solar installation. 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.

In this paper, we use the Mean Absolute Percentage Error (MAPE) as a statistical metric to measure the 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 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:

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

Here, A_t and P_t are the actual and the predicted value at time t. However, problems can occur when calculating the MAPE value with a series of small denominators. To circumvent 'divide by zero' problem, we change the denominator in the original formula from A_t to $A^{'}$, which is the average value over time t. Further, MAPE values calculated this way are insensitive to inclusion of nighttime actual and predicted values as both would be zero. Thus, we use the below formula

to report prediction errors in this paper.

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

3.1 Machine Learning Prediction Model

Much of the prior work has focused on predicting solar irradiance from weather forecasts, and then using the irradiance → power formula to predict the solar power. Since designing both the forecasts → irradiance model and irradiance → power model require historical observed irradiance, this approach is not scalable to millions of rooftop installations because the historical irradiance data is generally not available from these sites. So, instead, we focus on designing a prediction model that directly predicts solar power from weather forecasts for any solar installation in the country. 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. Later, we use a grid search technique to automatically determine the configuration parameters with a minimal amount of historical generation data.

As described in [23], 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 [23], we consider all weather parameters, 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 multiplying all weather parameters by the cloudless irradiance. Our intuition is 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 of the Sun, by multiplying the cloudless irradiance by the weather parameters our model also captures seasonal and diurnal variations in the Sun's position. Consequently, we formulate the power prediction regression model as:

$$P_t = f\left(S_t^{cloudless} \cdot \sum_{i=1}^{n} W_t^i\right) \quad (3)$$

Here, P_t and $S_t^{cloudless}$ are predicted power and cloudless irradiance, respectively, at time t, W_t^i is forecast of the i^{th} 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[23]. We apply a linear least squares regression technique on a training dataset to determine the function f. Linear least squares regression is a simple and commonly-used technique to estimate a value to be predicted from a set of variables, e.g., solar power, and a set of independent variables or predictors. The regression minimizes the sum of the squared differences between the observed solar power and the power predicted by a linear approximation of forecast parameters.

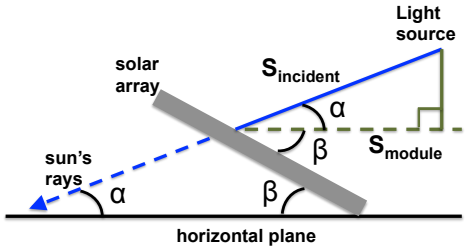


Figure 1. Tilt of panel (β) and solar elevation (α)

3.2 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 homeowners. Thus, to automatically learn these static parameters and generate site-specific prediction models we modify the above regression model as follows:

$$P_t = f'\left(S_t^{module} \cdot \nu_t \cdot \sum_{i=1}^{n} W_t^i\right) \quad (4)$$

Here, P_t is the predicted power, S_t^{module} is the perpendicular component of the cloudless irradiance on the panel, ν_t is the reduction factor due to the cell temperature, and W_t^i is forecast of the i^{th} weather parameter, all at time t. We learn the function f', for given values of static parameters, using the machine learning regression technique from the previous section. Since the perpendicular component of the cloudless irradiance depends on the panel's tilt and orientation, the Sun's altitude and azimuth, S_t^{module}, is expressed as:

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

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 (Figure 1). 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, cell temperature has a direct bearing on the power output from a photovoltaic panel. NOCT is defined as the cell temperature reached by open circuited cells in a module under ideal conditions (20°C, 800 Wm^{-2}), such that for every degree rise in the cell temperature above 25°C, there is a 0.5% reduction in the power output [25]. The cell temperature can be calculated from the ambient temperature for a given NOCT value as follows:

$$T_t^{cell} = T_t^{air} + \frac{NOCT - 20}{80} \cdot S_{module} \quad (6)$$

Here, T_t^{cell} and T_t^{air} are the cell and ambient temperature at time t. All units of temperature are in celsius. The unit for S_{module} is $mWcm^{-2}$. Hence, ν_t in Equation 4 is given by:

$$\nu_t = .005 \cdot (T_t^{cell} - 25) \quad (7)$$

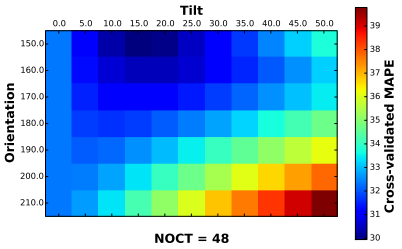


Figure 2. Prediction error for various values of tilt and orientation, and for a fixed NOCT value.

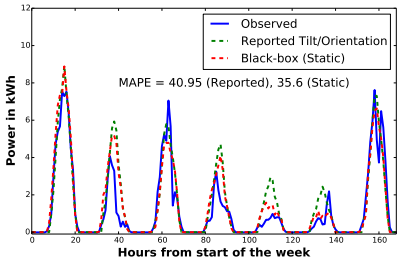


Figure 3. Grid search improves the prediction accuracy by more than 10% over a period of 7 days.

To automatically find the static parameters – tilt, orientation, and NOCT – for any solar panel installation, we employ a grid search technique. The grid search algorithm is a popular hyperparameter optimization technique. Here, we perform an exhaustive search in a three-dimensional hyperparameter search space guided by cross-validated MAPE on the training data. Figure 2 shows the MAPE for various values of tilt and orientation during the grid search for a fixed NOCT value of 48.

To realize the benefits of grid search we use the historical power data from a 10kW solar installation near Amherst, MA, and historical weather forecasts from the Forecast.io web service (http://forecast.io) for the same location. We use the first six months of data from July to December 2013 for training and the remaining six months from January to June 2014 for testing. We first discard erroneous values, and use linear interpolation to find missing values. We then generate the prediction model for the user-supplied tilt and orientation, and use the grid search algorithm to find the actual tilt and orientation. Figure 3 compares observed power, predicted power for the user-supplied tilt and orientation, and predicted power for the self-learned (using grid search) tilt and orientation for 7 days starting June 8, 2014. Figure 3 and Figure 4 (discussed later) are for visualizing the improvement in prediction; we provide comparisons over a longer timespan of 6 months in the evaluation section. As the figure indicates, the grid search automatically learns the static parameters, and thus provides better predictions than the model based on the user-supplied parameters, which are generally given by visual inspection and are often inaccurate.

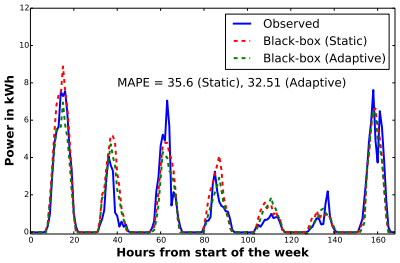


Figure 4. Using past information improves the prediction accuracy by more than 8% over a period of 7 days.

In summary, SolarCast automatically learns static configuration parameters such as tilt, orientation, NOCT, 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 form the site.

3.3 Adapting to Dynamic Factors

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 dynamic factors. To compensate for prediction errors due to the dynamic factors we add a new feature P_{t-24}^{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:

$$P_t = f'\left(S_t^{module} \cdot \nu_t \cdot \sum_{i=1}^{n} W_t^i + P_{t-24}^{output}\right) \quad (8)$$

Here, P_t is the predicted power, S_t^{module} is the perpendicular component of the cloudless irradiance on the panel, ν_t is the reduction factor due to the cell temperature, and W_t^i is forecast of the i^{th} weather parameter, all at time t. Since P_{t-24}^{output} is a dynamic parameter that changes every day, we retrain the model (and get a new function f') every day for the next day prediction. Note that running the grid search algorithm is not required every day because the grid search technique is used to learn the static parameters, which do not normally change for a particular installation. So, once we learn the static parameters of a site, using techniques from the previous section, we always use those values in its dynamic predictor (Equation 8).

To see how our prediction model leverages past information to account for errors due to dynamic factors we use the same dataset as above for training and testing. For a fair comparison with the static grid search model discussed above, we consider the a moving six months training period for the adaptive model. In other words, we retrain the adaptive model daily with the past six months of data. Further, we use the static parameters learned from above for the adaptive

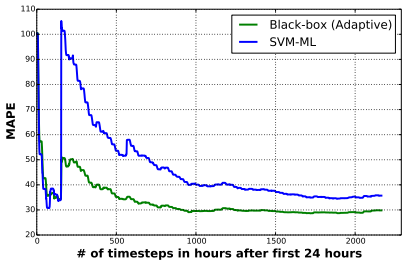


Figure 5. Prediction error for the online version of SolarCast's adaptive model and SVM-ML model.

model. As Figure 4 indicates, using past information improves the prediction accuracy. The MAPE value reduces from 35.6% (for grid search based static model) to 32.51% (for adaptive model) for the same duration.

In summary, SolarCast's dynamic model can learn dynamic environmental factors, such as dust, pollen, snow, etc., that vary seasonally or irregularly, and automatically corrects the model to account for effects due to the dynamic factors.

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 recording power generation since deployment. Further, not all homeowners install sophisticated monitoring devices, which can store data for several months or years. In such cases, SolarCast should be able to generate prediction models, albeit poor in the beginning, with as little as one or two days of historical data, and should keep retraining (and refining) the models as and when it gets new data from the sites.

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. To see how soon our online approach achieves 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 5, the online model rapidly converges, within 10%, to the static model within one month. This illustrates that any new or old installation in the country 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 [23] 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 improvement in MAPE is more sluggish. Also, notice MAPE for both approaches stabilizes with more data.

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.

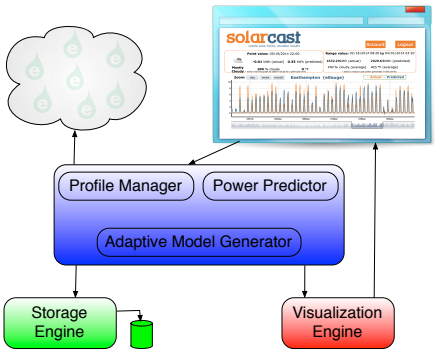


Figure 6. SolarCast Web Service Architecture

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 time scales ranging from an hour to a few days. Unlike prior services [19], 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 – (a) Profile Manager, (b) Visualization Engine, (c) Predictive Model Generator, (d) Power Predictor, (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 per-profile basis. A user can have multiple installation profiles, and a profile may be associated with multiple users.

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

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 histor-

ical/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.

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.

The power predictor is called when a user sends a request to generate 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. It calls the visualization engine to format the results and display that 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 [6], an open source web application framework written in Python, to build the SolarCast's web service. The visualization engine uses dygraph [7], 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. *scikit-learn* is an open source machine learning library for Python. In addition, we use libraries – SciPy, NumPy, Pandas – from the SciPy stack [22] for data processing. To store users' profiles, prediction models, and dataset we use SQLite, a lightweight disk-based relational database management system.

Since sensors used by many households report power readings in their local time zone, accounting for the daylight saving becomes difficult in the prediction model. For this purpose, we convert local time readings to standard unix time using the Python pytz library [21] that automatically handles the daylight saving issues. To get weather forecasts for any location we use the Forecast API from Forecast.io[8]. 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, 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 [20] that implements the Masters' algorithm.

5 Evaluation

We evaluate our black box prediction model on a large dataset of 118 solar installations spread across 16 different states in the US. The dataset consists of 116 small rooftop installations (four installations of 50-150kW capacity, three of 20-50kW capacity, and the rest of 5-20kW capacity each)

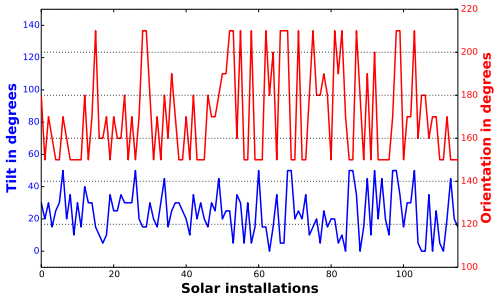


Figure 7. Tilt and orientation found using the grid search for each solar installation.

and 2 medium-sized solar farms of around 1MW capacity each. Each dataset contains the location – latitude and longitude – and historical power readings collected for twelve months using energy meters whose accuracy is discussed here [3]. We use first six months of data for training and last six months for testing.

We first learn the configuration parameters – tilt, orientation, and NOCT – for each site using the grid search algorithm. Next, we use the site-specific configuration parameters in our black box models (both static as well as adaptive) for each site. For each site, the static model, once generated over the training dataset, does not change over the testing period (of six months), where the adaptive approach recreates or refines the model every day, over past six months of historical data, to predict the next day 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) with linear kernel [23]. As opposed to predicting irradiance, we directly predict power based on weather forecasts for each day.

5.1 Learning Configuration Parameters

First, we use the grid search algorithm to learn configuration parameters – tilt, orientation, and NOCT – for each site. We vary the tilt between 0° and 50° in step of 5°, the orientation between 150° and 210° in step of 10°, the NOCT between 20°C and 60°C in step of 5°, and find the values that minimize the average MAPE over the training dataset. We randomly assign a number between 0 and 118 to each site. Figure 7 shows the tilt and orientation for each site (X-axis represents the site number). As the figure shows, 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

In this section, we compare the prediction accuracy of three models – our black box static model, our black box adaptive model, and the SVM-based prediction model from [23] – for each site. We use MAPE to measure the prediction accuracy for each model. Figure 8(a) plots MAPE for all three models for 116 rooftop installations, while Figure 8(b) plots the same for two open-space medium sized installation, each with 1MW capacity. Both figures show that the dynamic approach adapts to the dynamic parameters, such as dust, leaves, or pollen, and performs slightly better

(∼3-5%) than the static model that only learns the static configuration parameters and is oblivious to the dynamic factors. For both graphs, the SVM-ML model performs worse because it lacks important features, such as the cloudless irradiance, which accounts for the sun's elevation and azimuth, and it does not find the actual values of configuration parameters, which we find using the grid search algorithm.

5.3 Case Study: SolarCast in Smart Homes

In this section, we explore how households can leverage our black box prediction model to schedule elastic loads and reduce electricity bills. To maximize green energy penetration homeowners can schedule certain elastic loads, such as plug-in electric vehicles, washing machines and dryers, to run when solar energy is abundant. To experiment with sunny scheduling of the dryer in a smart home we choose a house located in the state of Massachusetts. The 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 and 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 assumptions 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 minimum load from the grid is drawn based on solar predictions that have been learnt 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 that multiple loads are not scheduled during the same timeslot. Excess power for timeslot j, is obtained by subtracting the electricity drawn from the grid to 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 prediction techniques to schedule elastic loads and reduce their electricity bills. As one example, a smart home reduces its grid energy demand by 40% by providing a little flexibility (of ∼5 hours) to startup time of a dryer.*

6 Related Work

Prior work on solar forecasting focus on predicting solar power for a particular solar panel installation or a few installations only. 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, which are often unknown for a typical rooftop installation. Lorenz et al. [14] and Huang et al. [10] 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. [26] use a neural network model to forecast solar irradiance; they then use a site-specific irradiance → power model to forecast power generation.

Tao et al. [25] 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 6:00 AM to 6:00 PM. To predict power at any arbitrary time it further requires additional input nodes with adequate training. These restrictions exist for [5] that uses Elman Neural Networks with the input layer very similar to that of [25]. Mandal et al.[15] 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 [23, 9, 13] 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.

7 Conclusion

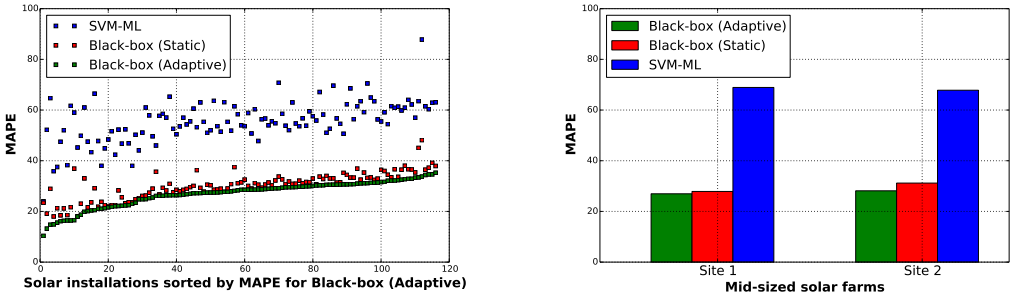
In this paper, we present a black box approach for forecasting solar power generation. Our black box model only needs the location and minimal historical data from any solar panel installation to design a custom site-specific prediction model. We evaluate our model for 118 solar installations, spread across 16 states in the US, and show that our grid search technique automatically learns static configuration parameters for each site. Further, 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 (average MAPE of 27% for all sites) compared to the average MAPE of around 50% for a prior machine learning based prediction model. We also present an application case study to show 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%.

8 Acknowledgements

This research was supported by NSF grants CNS-1405826, CNS-1253063, CNS-1143655, CNS- 0916577, and the Massachusetts Department of Energy Resources.

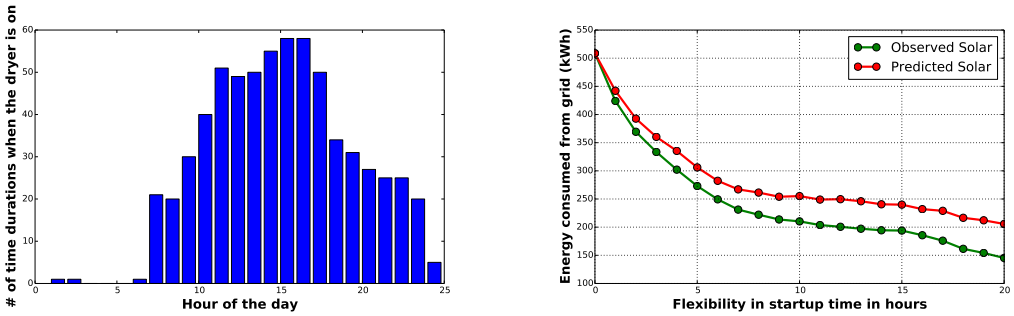
References

[1] 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.



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

Figure 8. Prediction error for various prediction models for each site over 6 months.



(a) (b)

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

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

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

[4] J. Barnes, T. Culley, R. Haynes, L. Passera, J. Wiedman, and R. Jackson. Best Practices in State Net Metering Policies and Interconnection Procedures. Technical report, Freeing the Grid, November 2013.

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

[6] Django Project. https://www.djangoproject.com/.

[7] Dygraph. http://dygraphs.com/.

[8] Forecast io. https://developer.forecast.io/docs/v2.

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

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

[11] Inverter, Storage and PV System Technology Industry Guide 2014. http://www.pv-system-tech.com/, 2014.

[12] S. Kaplan. Power Plants: Characteristics and Costs. Technical report, Congressional Research Service Report to Congress, 2008.

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

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

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

[16] S. Marcacci. 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.

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

[18] G. Petlin and K. Wu. California Net Energy Metering Ratepayer Impacts Evaluation. Technical report, California Public Utilities Commission, October 2013.

[19] PVwatts. http://rredc.nrel.gov/solar/calculators/pvwatts/version1/.

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

[21] PYTZ Python Library. http://pytz.sourceforge.net/.

[22] Scipy Stack. http://www.scipy.org/stackspec.html.

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

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

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

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