Solar Energy Prediction for Constrained IoT Nodes Based on Public Weather Forecasts

Frank Alexander Kraemer, Doreid Ammar, Anders Eivind Braten, Nattachart Tamkittikhun and David Palma
Department of Information Security and Communication Technology
NTNU, Norwegian University of Science and Technology
Trondheim, Norway
{kraemer, doreid.ammar, anders.e.braten, nattachart.tamkittikhun, david.palma}@ntnu.no

ABSTRACT
Solar power is important for many scenarios of the Internet of Things (IoT). Resource-constrained devices depend on limited energy budgets to operate without degrading performance. Predicting solar energy is necessary for an efficient management and utilization of resources. While machine learning is already used to predict solar power for larger power plants, we examine how different machine learning methods can be used in a constrained sensor setting, based on easily available public weather data. The conducted evaluation resorts to commercial IoT hardware, demonstrating the feasibility of the proposed solution in a real deployment. Our results show that predicting solar energy is possible even with limited access to data, progressively improving as the system runs.

ACM Classification Keywords
C.2.3 Network Operations: Network management; I.2.8 Problem Solving, Control Methods, and Search: Plan execution, formation, and generation; I.2.9 Robotics: Sensors

Author Keywords
Internet of Things; Machine Learning; Solar Energy; Constrained Nodes; Weather Forecasts

INTRODUCTION
Energy harvesting provides a sustainable source of power for Internet of Things (IoT) nodes and also simplifies their deployment. The downside is that their performance often changes considerably over time. In case of solar energy, variations occur throughout the seasons and depend on climatic parameters like weather, temperature, solar irradiance, hourly solar angle, season and geographical location [26, 27]. Tilt angle and orientation of the solar panel [8], and other effects like shadows [15, 18], also influence the output that can be produced by a stationary photovoltaic (PV) panel. Rechargeable batteries dampen fluctuations through energy buffering, but only to some degree. For instance, within a project involving the deployment of solar-driven nodes to measure greenhouse gas emissions [3], sensor devices worked properly during summer, but failed during winter when sun exposure was lacking.

Data quality and energy consumption are closely related [14]. Instead of abruptly running out of power, nodes should adapt their operation [11]. Sensor devices can for instance average over several measurements to reduce the number of transmitted messages and save power [17]. On the other hand, to increase accuracy and adapt to sudden and unexpected changes, sensors may increase their sampling rate at the cost of increased energy consumption [5]. Additionally, systems consisting of several nodes may also balance the load between them, improving energy consumption while providing a good overall sensing coverage [28].

Node failure due to lack of energy is one problem; another is to not use enough of the available energy to gain better data. When a battery is fully charged, all solar energy that exceeds the consumption of a node is wasted [13]. Instead, the sensor node could have consumed more energy and thereby acquired more data or data with higher accuracy. For that, sensors need to be aware of their current and future energy budget and plan ahead so that they can operate optimally. Due to the scale of IoT, such optimizations must happen autonomously. And since sensors are placed into heterogeneous and changing environments, optimizations must happen continuously and for each sensor individually. One method of choice is machine learning to predict the energy budget of a sensor node. An important component in such a prediction is the solar energy, which depends on features such as the position of the sun relative to the solar panel, the atmosphere and other effects.

Forecasting the available solar power with different time horizons is already part of the operation of solar power plants, to predict fluctuations in energy production and quickly compensate for sudden changes. These deployments are usually large in scale and can look back on a long history of available data of dedicated sensor information, for instance by observing the clouds in the sky by cameras, and even allow for manual supervision [9]. In an IoT setting, such a high de-

IoT 2017, October 22–25, 2017, Linz, Austria

 ISBN 978-1-4503-5318-2/17/10…$15.00
DOI: https://doi.org/10.1145/3131542.3131544

gree of instrumentation and supervision is unrealistic. Sensor nodes are placed at many different locations, and requiring specific orientation towards the sun would complicate the deployment process. Adding more instrumentation to nodes (like irradiation measurement or sky cameras observing the cloud coverage) would make them more expensive and complex. Instead, systems should be based on off-the-shelf sensor nodes and require only minimal instrumentation and setup.

In this paper, we present the results of an experiment on a prototype designed to investigate how and to which degree machine learning algorithms can be used to predict the solar energy budget for sensor nodes, based on easily available input data. In detail, our contributions are the following:

- An explanation of effects in embedded systems that need to be taken into account with a set of data preparation steps to gain good training data.
- A study of which features are most effective as input for solar energy prediction.
- A study of different machine learning algorithms and how they score as predictors.

One of the most crucial factors for success in machine learning is the availability of data. Due to the heterogeneity of environments into which sensor devices are deployed, this may be a problem. We therefore also study how the approach develops over time, starting with no initial data.

The focus of this paper is the prediction of solar energy. Nevertheless, we also outline the planning algorithm that selects the proper sensor operation mode, since it gives context to the solar energy prediction. Our approach also takes the constraints typical in IoT into consideration: Since machine learning is executed on a server as part of the device management, sensor nodes only require minimal computational effort. As communication we use the constrained LoRaWAN protocol.

RELATED WORK

Machine learning is already widely used in big scale forecasting for large solar-power farms. Weather data from numerous sources are blended with several models and methods via post processing, in order to produce the most accurate solar-energy forecast possible [9]. This requires high computational power and access to large amounts of data. However, these systems use highly specialized models tuned to the specific location of the solar farm. For more distributed energy resources, where small scale photovoltaic panels are connected to a smart grid, making an accurate prediction of the expected output is also important. These are used to forecast and plan the total distribution of the produced energy over the entire energy grid [24, 27]. In contrast to the solar farms, this needs to be done with more generalized models and less fine-grained weather data. The power produced by a PV-panel highly depends on the irradiation reaching it. Shi et. al [19] propose a forecasting model to predict the output of PV-panels based on a classification of the weather. They show that weather conditions, clouds, solar angle, and season are factors that must be taken into consideration when predicting the energy budget for a solar-energy harvesting device.

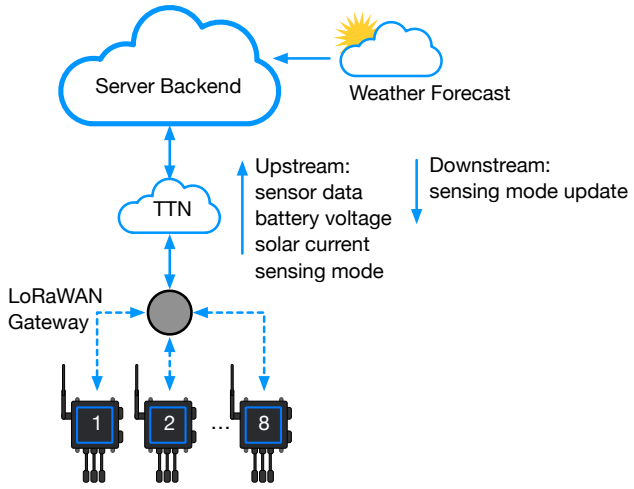

Figure 1. Overview of the system.

Kansal et al. [12] show how power management is inherently different for a node using energy-harvesting compared to one powered by battery only. This is due to the variability of available energy and because conventional energy optimization methods are not always optimal in an energy-harvesting scenario. As such, it is important to adapt the workload to the amount of energy that can be harvested. They demonstrate how a closed-loop electronic circuit can be used to predict and plan the energy budget to achieve energy-neutrality. However, they do not consider the usage of weather forecasts in their energy planning algorithm, since they assume that the expected energy production is typically the same on a given time for consecutive days. This, however, does not apply to many parts of the world, where the weather condition can shift from day to day, and even from hour to hour. Hsu et al. [10] propose a modified power management method that is taking unstable and uncontrollable conditions into consideration, using reinforcement learning. They claim their method gives a performance increase of 2.3 % in summer time.

Constrained devices that scavenge for energy in a location where weather conditions are shifting, need to be able to predict their energy budget to keep a steady battery level. Szydlo et al. [22] propose a concept of a two-stage predictive power-adaptation method that uses weather forecast services to plan how much energy it is possible to harvest from the wind in the near future. Their aim is to create a power management system that address the problem of optimal control of battery neutrality under shifting weather conditions, and that at the same time can guarantee a satisfactory level of functionality. Based on these plans, they propose to change the energy consumption of the devices by switching between four operation modes, which use different amounts of energy. This is a similar approach to ours. However, their results are based on values gathered from a simplified test unit and then run in simulations, while we are using an off-the-shelf sensor station and have built a prototype for running the test and measure the actual values.

Figure 2. Testbed with five of the eight sensor nodes.

SYSTEM OVERVIEW

Figure 1 shows an overview of the system. It consists of sensor nodes that communicate via gateways with a backend server.

Sensor Nodes

We deployed eight Waspmotes [16] from Libelium, shown in Figure 2. They are based on the 8-bit Atmega1281 microcontroller. Each sensor node is coupled with sensors to measure CO_2, temperature, pressure and humidity. Some sensors also measure particle matters (PM) and NO_x. However, since, we focus on the energy management, only the energy consumption of the sensors is relevant in the following. The CO_2 sensors, for instance, include a heater to correctly capture the gas density at a specific temperature. Therefore, they require a considerable amount of energy when sensing.

Each sensor node is powered by a lithium-ion polymer (LiPo) battery with the capacity of 6600 mAh and a maximum voltage of 4.2 V. The battery is connected via a charging controller to a solar panel, which can provide a current of up to 330 mA. For this experiment, all solar panels face the same direction. The controller protects the battery from overcharging, which has some implications for our data as explained later.

The sensor nodes periodically execute sensing cycles, i.e., they wake up, make some measurements, send the results to the LoRaWAN gateway and then go into sleep mode again. To adjust their behavior, sensor nodes can be configured using discrete *sensing modes* [23]. Each sensing mode assigns specific values to the length of the sleep cycle and the number of measurements within each sensing cycle. The sensing modes are designed so that a lower sensing mode yields less frequent measurements and less samples per sensing cycle, and accordingly uses less energy. Higher sensing modes provide more and better data and use more energy.

Network

For the wireless connection, we use LoRaWAN (868 MHz and 433 MHz ISM band) [20]. The corresponding gateway is deployed 500 meters away and connected to The Things Network (TTN, [2]). Before put into deep sleep, sensor nodes transmit the necessary data required by the backend server to TTN's gateways via the LoRaWAN uplink channel. This is the actual sensor data together with the energy-related data i.e., the battery voltage, the incoming solar current and the sensing mode. From TTN, our server fetches the data of all sensor nodes at regular intervals, as explained later. LoRaWAN also provides a downlink channel from the server towards the sensor nodes. We use this channel to update the sensing mode in the sensor nodes as a result of the energy planning. The fair access policy of LoRaWAN and TTN limits the uplink transmission to a 30 seconds airtime per day per node, which corresponds to roughly 647 bytes per day, given the spreading factor of 7. The downlink is even more restricted, with 10 messages per day per node.

Server Backend

The server backend collects all data and has the task to determine the optimal sensing mode for each sensor node. Figure 3 provides an overview. The server operations are coarsely structured into learning, predicting and planning.

The server repeatedly collects the data sent via LoRaWAN to The Things Network (TTN) from their servers **(1)**. This raw byte data is decoded so that the individual data fields can be stored as files in CSV format. The server also collects weather forecast data **(2)**. Both the weather and the sensor device data are combined into training and testing data for the various machine learning methods **(3)**, explained later in detail.

For the prediction part, there are three machine learning modules. The first one **(5)** predicts the solar energy output based on the weather data and time, and is the main focus of this paper. The other two modules are used to predict the expected battery level based on the incoming solar current **(6)**, and to predict the energy consumption for a sensor node given its sensing mode **(7)**.

The prediction modules contain different machine learning algorithms, detailed later. They are trained by corresponding modules. This means, for instance, if **(5)** is a neural network, **(4)** encapsulates the execution of the backpropagation algorithm to train it.

The planning module **(8)** uses the prediction modules to simulate several potential energy budgets for a day, given different sensing modes. These potential budgets are then evaluated, which leads to the selection of the best sensing mode for each sensor node, which is then sent back to the them as a sensing mode update, via LoRaWAN.

DATA ACQUISITION

Table 1 lists the data required for learning.

Table 1. Overview of input data

Energy data	battery voltage, solar charge current, sensing mode
Weather data	forecast time, production time, location, temperature, wind speed, wind direction, pressure, humidity, cloudiness (high, medium, low, total), fog, dewpoint temperature, precipitation, weather symbol
Sun position	zenith, azimuth

Energy Data from Sensor Nodes

The firmware of the Waspmotes provides access to the voltage at the battery and the current arriving from the solar panel. The latter is a good indicator for the available solar energy, and will be the value to be predicted by our algorithms in the next section. There is one caveat with the current from a solar panel:

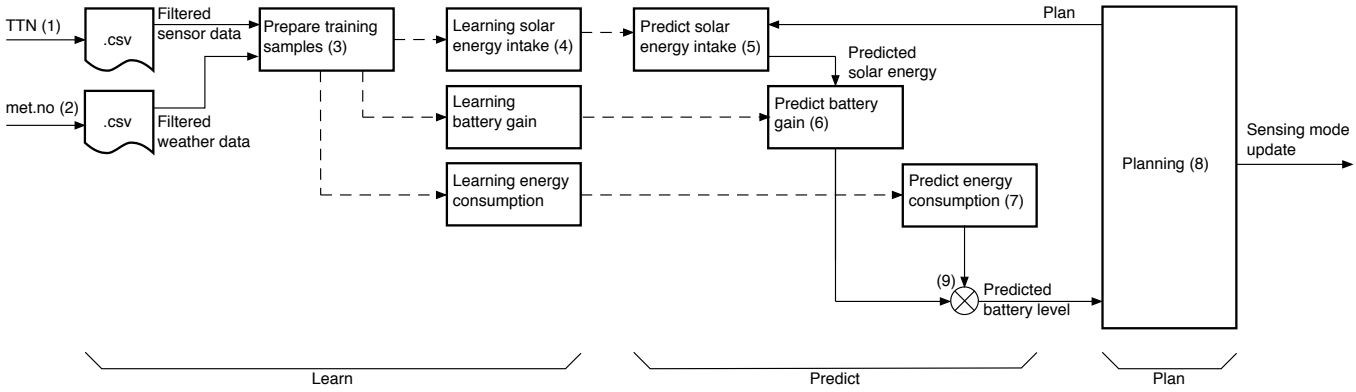

Figure 3. Overview of the operations in the server backend.

It only flows if the Waspmote also consumes energy. We will later see which consequences this has for the data preparation. However, measuring the current from the solar panel is still the best indication for the available solar energy. In contrast, only measuring the open-circuit voltage at the solar panel is less useful. Solar panels are usually non-linear in terms of sunlight intensity versus output open-circuit voltage [4]. This means that the open-circuit voltage is an indication if the sun is shining, but not very precise regarding the available energy.

Weather Forecast and Sun Position

The weather forecast data is collected from the Norwegian Meteorological Institute via their publicly accessible API [1]. Forecasts are published at irregular intervals, about three times a day. The different data fields are listed in Table 1. Each weather forecast has a production timestamp, i.e., the time when it was created. The weather forecast includes predictions for several points in time in the future, each labelled by the forecast timestamp. For each forecast timestamp, the report lists several numerical weather values. Cloudiness is provided at three different levels (low, medium, high) as well as an aggregated value (total). The weather symbol is a discrete value between 1 and 50 that encodes a weather scenario. For instance, sun is encoded as 1, rain as 10 and snow as 13.

The position of the sun is expressed in terms of two angles, azimuth and zenith. These can be calculated on the server based on the location of the deployment and the time, i.e., this data does not need to be collected.

DATA PREPARATION

The effectiveness of the machine learning algorithms depends to large degrees on the preparation of the input data. This preparation requires knowledge about the domain of the system, i.e., the electrical properties of batteries, solar panels, the charging controller and some knowledge about the sun and weather forecasts.

Preparing Battery Level Data

For Li-ion/LiPo batteries, the battery level (also called *state-of-charge*, *SoC*), and the voltage at the battery have a relationship that can be approximated by a linear model as shown in Figure 4. It is therefore possible to estimate the battery level as percentage from 0 to 100 based on the measured voltage at

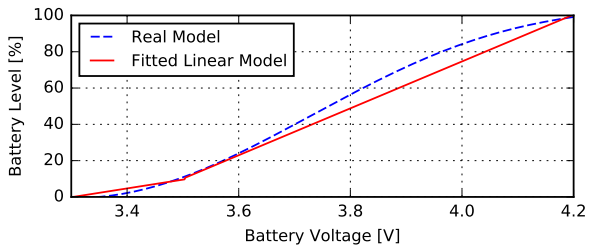

Figure 4. The relationship between the voltage and battery level.

the battery. For our experiment, we use the approximation provided by Libelium as part of their firmware [7]. The conversion from battery voltage to battery level in percent is only an issue of understanding the data, since it is more convenient to look at a percentage ($0 \ldots 100\,\%$) than at a voltage ($\approx 3.3 \ldots 4.2\,\mathrm{V}$) when talking about the state of a battery. For the actual prediction this conversion has little impact as long as it is done consistently. In our case, we derive the battery level from two approximated linear functions of the battery voltage illustrated by the two connected solid lines, as opposed to the unknown real model shown as dotted line in Figure 4, which represents the general curve of Li-ion/LiPo batteries [6].

Preparing Current Data from Solar Energy

Figure 5 shows the solar current throughout three days. April 7th is a day with extremely varying weather, which makes the solar current vary over the day. April 9th is a consistently cloudy day, and April 12th is a very sunny day. The curve of the sunny day shows a steep slope because of the sunrise at around 6:00. However, at 8:45 the solar current drops to zero. This is due to the charging controller. When the battery level is rising above 98 % or 99 % (depending on the node), the charging controller switches off charging and the Waspmote is powered from the battery only, even if solar energy is available. The charger switches back into the charging state once the battery voltage falls below $\approx 4.07\,\mathrm{V}$, which corresponds to $\approx 83.7\,\%$ of battery level. This explains why there are periods in which the solar current is zero despite the shining sun.

The machine learning algorithms need to know about this charging behavior; otherwise, they would be confused by sunny conditions that seemingly do not lead to the expected

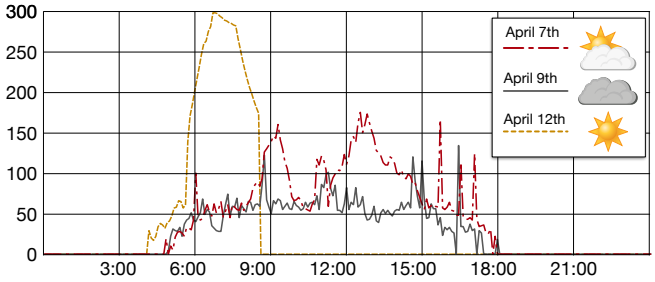

Figure 5. The solar current on three days with very different weather conditions.

solar current. Adding data about the charging state is part of the data preparation of the solar current. Since Waspmotes do not provide any data regarding the state of the charger, we have to guess it from the charging current and the battery level. For this, we look for the point when charging stops, and observe the battery level at the same point for all days of a node. Then we select the highest battery level as the full battery threshold for that node. The thresholds for all the nodes are used in the data filtering program to assign battery full state to data points. After every data point is marked with states, those with the state *battery full* are removed.

Aggregation of Weather Forecast

Each forecast is provided as a table, where each row describes the expected weather at specific points in time. For the close future, this interval is one hour. For predictions further in the future, these intervals are increasing to up to six hours. Each row has therefore two timestamps: the time for which the weather is predicted (forecast time), and the time when the forecast was produced (production time). Since the forecasts cover overlapping times, the data preparation selects for each time the forecast where forecast time and production time have the shortest distance. This means that the most up-to-date forecast is selected.

ENERGY BUDGET PLANNING

The planning algorithm is only sketched in the following since it is not the focus of this article, and rather provides context for the machine learning and prediction of the solar power current. This algorithm simulates the development of the battery level, given that the node executes a specific sensing mode. As input it uses the current battery level, the sensing mode and the weather prediction. The planning algorithm simulates the time ahead in intervals, for which we selected 30 minutes. It first uses the machine learning module for the solar power prediction (see Figure 3 **(5)**) to predict the expected solar current, detailed in the next section.

From the predicted solar current, the planning algorithm needs to estimate how the battery will develop within the simulated 30 minute interval. This depends on two components: (1) how much energy the sensor node consumes in the given sensing mode, and (2) the energy added from the solar panel. These two components can be estimated by two different modules, and then added together, as done in Figure 3 **(9)**).

- To analyze the energy consumption of a given sensing mode (Figure 3 **(6)**), we select periods during the night in which the solar energy is zero. During these periods, the battery level decreases. We analyze the slope, averaging over several nights, which gives a good indication of the energy consumption of the currently executed sensing mode.

- To come from the solar intake to the increase in battery level, we need to determine a simple factor that calculates the increase in battery level from the average solar current for a given time interval. We currently estimate this factor manually by comparing simulated data with real ones, but we foresee also here automated statistical methods.

The plans for the different sensing modes are compared with each other, based on a utility function that penalizes an empty battery or wasting solar energy. The planning algorithm then selects the sensing mode that leads to a plan with the best utility, and updates the sensing mode of a sensor node accordingly, using the downstream LoRaWAN channel.

SOLAR POWER PREDICTION

The biggest unknown influence on the energy budget is the availability of solar energy, which varies considerably between days based on the weather. The following section shows the prediction accuracy of the solar current of different machine learning methods, taking the weather forecast and solar angles as input features.

Training and Test Data

In our experiment, we are also interested in the bootstrapping problem, that means, how the accuracy of the prediction will develop over time when we start with no data at all. This corresponds to a realistic scenario of a fresh deployment of sensors that has not yet observed any data at all. We simulate therefore how the training data as well as the algorithms will develop over time. For each day d_n with $n \in \{0,\ldots,60\}$ for which we have collected data, we use all data from the previous days $d_0,\ldots,d_{n-1}$ as training data to create a new model R_n. Data of day d_n serves as test data. In our context, it is important to take data from entire days as test data, and remove those entire days from the training data, to avoid data leakage. Simply taking out 20 % of random values over a longer period is not good enough. The relatively slowly changing conditions would lead that the training data to contain very similar samples as there are in the test data. Instead, our method of using an entire day as test data and only using data from previous days as training data corresponds to a causal way that gives realistic results.

Figure 6 shows at its top the number of data samples in the training and test sets (on a logarithmic scale). The lower curve shows the samples from each day, which are used as test data for that day. Its minimum is on the 15th of April, where there was very little training data because the batteries of all nodes were fully charged. The upper curve shows the training data used for building the model on that day. Since the training data for day d_n is the collective test data of all days $d_0,\ldots,d_{n-1}$ before it, the number of training samples monotonously increases.

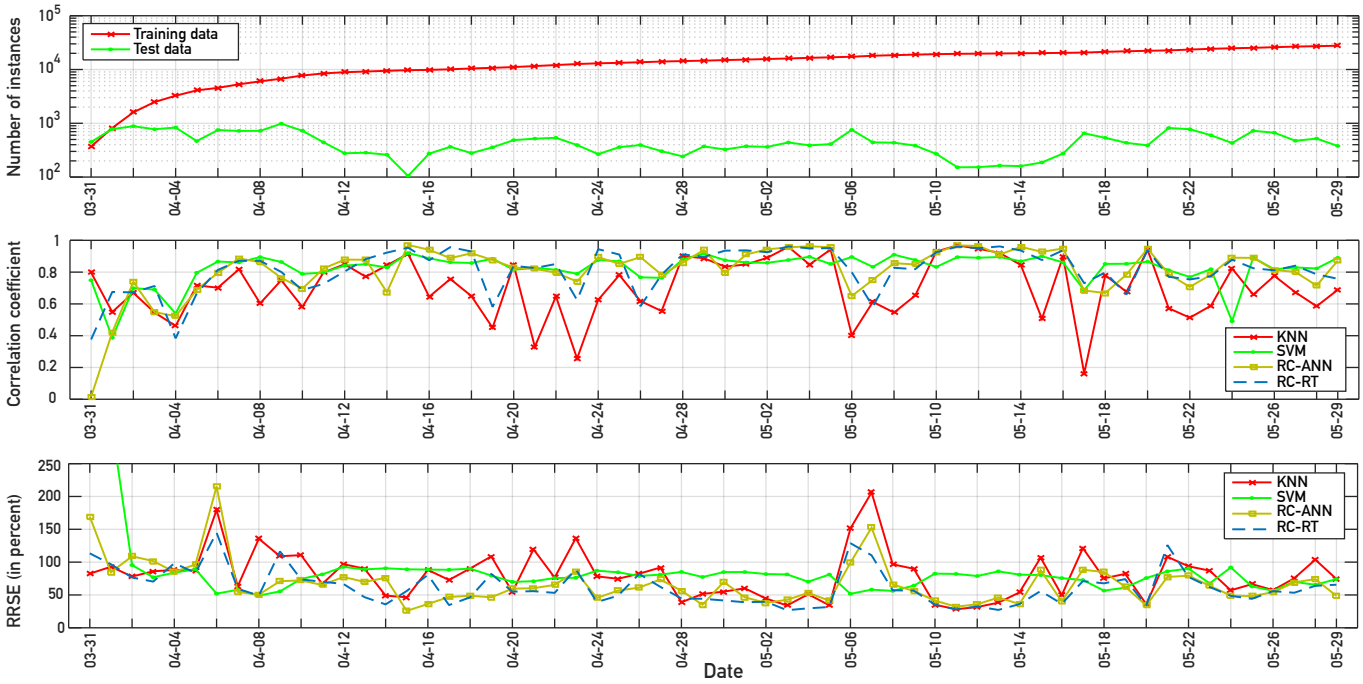

Figure 6. Top: Number of instances of training and testing data. Middle and bottom: Correlation coefficient and RRSE (root relative squared error) for the models built for each day.

Attribute Selection

To improve the accuracy of solar power prediction, we used the *Ranker* method in [25] for attribute selection to rank attributes and to perform attribute selection by the removal of redundant and irrelevant attributes. As a result, the selected list of features, sorted by ascending attribute rank-order, is the following: zenith, azimuth, low clouds, high clouds, temperature forecast, medium clouds, symbol.

Machine Learning Algorithms

In our work, we have selected a subset of widely-used ML algorithms, namely *k-nearest-neighbor* (*k*-NN), *Support Vector Machines* (SVM), *Artificial Neural Networks* (ANN), *Random Tree* (RT) decision tree learner and *Random Committee* (RC), which are briefly described hereafter. For more details, please refer to [25].

- *k*-nearest-neighbor (*k*-NN) is an instance-based lazy learner that compares the value to predict with existing values gathered from the *k* closest training instances (neighbors) using a distance metric. This algorithm requires all the training instances to be kept in memory and does not produce any model.

- Support Vector Machines (SVM) is a technique that builds cutting hyperplanes, which separate the data in an optimal manner.

- Artificial Neural Networks (ANNs) consist of an interconnected network of nodes (artificial neurons). Each node maps complex relationships between inputs and outputs using weights learned iteratively over the training data by the backpropagation learning algorithm.

- Decision Trees are prediction models in the form of a tree graph that are built by binary splitting the set of data using a recursive greedy search algorithm. We selected the *RandomTree* decision tree algorithm that considers a given number of random features at each node.

- Random Committee is a meta technique that can be applied on other algorithms in order to turn them into more powerful learners. It builds an ensemble of base algorithms and averages their predictions in a way to avoid overfitting and reduce the variance of the algorithm output. This technique makes sense if the base algorithm is randomized. In our work, *Random Committee* works only for ANN and *Random Tree* since the base of these algorithms is randomized, which is not the case for *k*-NN and SVM.

Building Machine Learning Models

For each of the tested ML algorithms, we built models using the entire training set by means of the Waikato Environment for Knowledge Analysis (Weka) [25]. In this work, we have conducted several tests to select a good tuning for each algorithm. However, tuning these models further is beyond the scope of the paper.

EVALUATION AND DISCUSSION

The middle of Figure 6 shows the correlation coefficient for the various models on each day. This coefficient measures the statistical correlation between the actual values of solar power and the predicted values. It ranges from -1 for perfect negative correlation, through 0 when there is no correlation, to 1 when the results are perfectly correlated. The bottom of Figure 6 shows the root relative squared error (RRSE), which is one of the error evaluation measures described in [25].

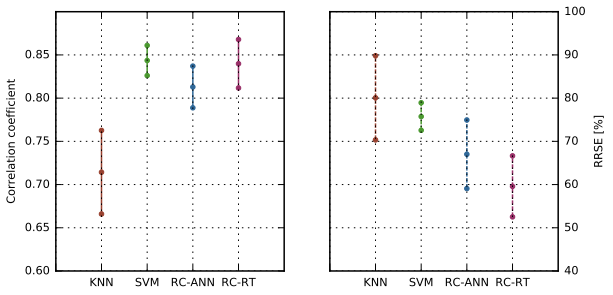

Figure 7. Average correlation coefficients and RRSE for the models.

Note that the best numerical prediction model is still the best no matter which error measure is used. Figure 7 shows the average correlation coefficients and the average RRSE for all models, with a 95 % confidence interval.

The accuracy of the prediction depends on several factors:

- The accuracy of the weather forecast, this means to which degree the weather forecast corresponds to the actual weather at the sensor node.
- New weather situations that have not been observed before.
- Missing data, which can happen either due to node failures or transmissions problems.
- Long periods with fully charged batteries in many nodes at the same time. In these cases, the system does not learn anything about the solar energy available.

The RRSE of most models spike on the 6th and 7th of May On these days, the pattern of reported solar energy was significantly different from the two weeks before. Revisiting Figure 5, we see that the most common pattern was similar to that on April 12th with a spike in charging and a fully charged battery. However, in some other days where we also registered an increased error, the intake is distributed over time and without any identifiable pattern. This could be due to a number of factors (i.e., weather forecast and battery level) and we believe with more data such variations will be less frequent.

Concerning the lack of learning data when batteries are fully charged, we can see this effect in the period following May 25th, when most batteries were full due to very good weather. This resulted in few samples for the solar intake, that were in addition taken in short periods between fully charged batteries, which is why they may not accurately represent the actual solar energy.

Quality of the Predictors

An overall analysis of the results for each predictor reveals their different properties. *k*-NN is not a good predictor. It is unstable, with the lowest correlation coefficient and highest RRSE. The other predictors are better. SVM is quite stable and its average RRSE is 76 % with a low standard deviation. RC-RT has the lowest RRSE and a high value of the correlation coefficient. In summary, SVM, RC-ANN and RC-RT are good predictors. They may even improve with further tuning of these algorithms.

Suitability of the Prediction

Since the learning process starts at the second day of the systems operation with only the data from day one, the prediction accuracy is understandably low during the first days. However, we see that it is improving and, for the methods other than *k*-NN, remains relatively stable. Based on our 60 days of observations, we argue that the predictions are already useful for a planning algorithm. The predictions rely on the accuracy of the weather forecast, which itself is prone to errors. In addition, the weather at the test site in Trondheim, Norway, is very volatile. Of course, one remedy would be to maintain a dedicated local weather station. However, with that we would make the deployment of IoT nodes more complicated; our intention is to only rely on easily available public weather forecast data.

CONCLUSION

We have presented an approach to predict the solar energy input for constrained IoT nodes based on numerical weather forecasts that are typically easily available. This allows for effective energy-budget planning, which is much needed for resource-constrained nodes. We have also observed that the choice of machine learning method matters. *k*-NN shows the biggest drop in accuracy on some days, while the other algorithms stay more stable.

Given that the predictions are based on weather forecast data, which itself contains uncertainties, a planning algorithm needs to take the accuracy of the prediction into account. It can for instance analyze confidence intervals, study best and worst cases and eventually select the most adequate strategy. In addition, we have observed that the behavior of the charging controllers used in off-the-shelf IoT nodes leads to less training data, since they cannot measure the available solar energy once the battery is full. This should have influence on the design of solar-driven embedded IoT nodes; they should be able to gain some insights on the available solar energy even when fully charged.

Acknowledgment

David Palma is funded by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Grant Agreement No. 699924, SINet, Software-defined Intermittent Networking project.

REFERENCES

1. 2017. Meteorologisk institutt. `https://api.met.no`. (2017). Accessed: 2017-06-06.
2. 2017a. The Things Network. `https://www.thethingsnetwork.org`. (2017). Accessed: 2017-06-06.
3. Dirk Ahlers, Patric Arthur Driscoll, Frank Alexander Kraemer, Fredrik Valde Anthonisen, and John Krogstie. 2016. A Measurement-Driven Approach to Understand Urban Greenhouse Gas Emissions in Nordic Cities. *NIK Norsk Informatikkonferanse* (Nov. 2016), 1–12.
4. M. Ashry and S. Fares. 2012. Electrical characteristic measurement of the fabricated CdSe/p-Si heterojunction

solar cell under radiation effect. *Microelectronics and Solid State Electronics* 1, 2 (2012), 41–46.

5. Gabriel Martins Dias, Maddalena Nurchis, and Boris Bellalta. 2016. Adapting Sampling Interval of Sensor Networks Using On-Line Reinforcement Learning. *2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)* (2016), 460–465.
6. Daniel Gandolfo, Alexandre Brandão, Daniel Patiño, and Marcelo Molina. 2015. Dynamic model of lithium polymer battery–Load resistor method for electric parameters identification. *Journal of the Energy Institute* 88, 4 (2015), 470–479.
7. David Gascon, Alberto Bielsa, David Cuartielles, and Yuri Carmona. 2016. Waspmote API v026 - WaspPWR.cpp. `http://www.libelium.com/api/waspmotev26/da/d2b/WaspPWR_8cpp_source.html`. (Dec 2016). [Source Code]. Accessed: 2017-06-06.
8. Michael Hartner, André Ortner, Albert Hiesl, and Reinhard Haas. 2015. East to west–The optimal tilt angle and orientation of photovoltaic panels from an electricity system perspective. *Applied Energy* 160 (2015), 94–107.
9. Sue Ellen Haupt and Branko Kosovic. 2015. Big Data and Machine Learning for Applied Weather Forecasts: Forecasting Solar Power for Utility Operations. In *Computational Intelligence, 2015 IEEE Symposium Series on*. IEEE, 496–501.
10. Roy Chaoming Hsu, Cheng-Ting Liu, Kuan-Chieh Wang, and Wei-Ming Lee. 2009. QoS-aware power management for energy harvesting wireless sensor network utilizing reinforcement learning. In *Computational Science and Engineering, 2009. CSE'09. International Conference on*, Vol. 2. IEEE, 537–542.
11. Hrishikesh Jayakumar, Kangwoo Lee, Woo Suk Lee, Arnab Raha, Younghyun Kim, and Vijay Raghunathan. 2014. Powering the internet of things. In *Proceedings of the 2014 international symposium on Low power electronics and design*. ACM Press, New York, New York, USA, 375–380.
12. Aman Kansal, Jason Hsu, Mani Srivastava, and Vijay Raghunathan. 2006. Harvesting aware power management for sensor networks. In *Proceedings of the 43rd annual Design Automation Conference*. ACM, 651–656.
13. Aman Kansal, Jason Hsu, Sadaf Zahedi, and Mani B Srivastava. 2007. Power management in energy harvesting sensor networks. *ACM Transactions on Embedded Computing Systems (TECS)* 6, 4 (2007), 32.
14. Victor Lawson and Lakshmish Ramaswamy. 2015. Data Quality and Energy Management Tradeoffs in Sensor Service Clouds. In *Big Data (BigData Congress), 2015 IEEE International Congress on*. IEEE, 749–752.
15. Buvana Lefevre, Stef Peeters, Jef Poortmans, and Johan Driesen. 2017. Predetermined static configurations of a partially shaded photovoltaic module. *Progress in Photovoltaics: Research and Applications* 25, 2 (2017), 149–160.
16. Libelium 2016. *Waspmote Datasheet*. Libelium. v7.0.
17. Clemens Moser, Lothar Thiele, Davide Brunelli, and Luca Benini. 2010. Adaptive power management for environmentally powered systems. *IEEE Trans. Comput.* 59, 4 (2010), 478–491.
18. Yousra Shaiek, Mouna Ben Smida, Anis Sakly, and Mohamed Faouzi Mimouni. 2013. Comparison between conventional methods and GA approach for maximum power point tracking of shaded solar PV generators. *Solar energy* 90 (2013), 107–122.
19. Jie Shi, Wei-Jen Lee, Yongqian Liu, Yongping Yang, and Peng Wang. 2012. Forecasting power output of photovoltaic systems based on weather classification and support vector machines. *IEEE Transactions on Industry Applications* 48, 3 (2012), 1064–1069.
20. N. Sornin, M. Luis, T. Eirich, T Kramp, and O. Hersent. 2015. *LoRaWAN Specification* (1 ed.). LoRa Alliance.
21. Sujesha Sudevalayam and Purushottam Kulkarni. 2011. Energy harvesting sensor nodes: Survey and implications. *IEEE Communications Surveys & Tutorials* 13, 3 (2011), 443–461.
22. Tomasz Szydło and Robert Brzoza-Woch. 2015. Predictive power consumption adaptation for future generation embedded devices powered by energy harvesting sources. *Microprocessors and Microsystems* 39, 4 (June 2015), 250–258.
23. Nattachart Tamkittikhun, Amen Hussain, and Frank Alexander Kraemer. 2017. Energy Consumption Estimation for Energy-Aware, Adaptive Sensing Applications. In *International Conference on Mobile, Secure and Programmable Networking (MSPN'2017)*.
24. Wayes Tushar, Bo Chai, Chau Yuen, David B Smith, Kristin L Wood, Zaiyue Yang, and H Vincent Poor. 2015. Three-party energy management with distributed energy resources in smart grid. *IEEE Transactions on Industrial Electronics* 62, 4 (2015), 2487–2498.
25. Ian H Witten, Eibe Frank, Mark A Hall, and Christopher J Pal. 2011. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann.
26. Amit Kumar Yadav and SS Chandel. 2014. Solar radiation prediction using Artificial Neural Network techniques: A review. *Renewable and Sustainable Energy Reviews* 33 (2014), 772–781.
27. Hong-Tzer Yang, Chao-Ming Huang, Yann-Chang Huang, and Yi-Shiang Pai. 2014. A weather-based hybrid method for 1-day ahead hourly forecasting of PV power output. *IEEE transactions on sustainable energy* 5, 3 (2014), 917–926.
28. Hongseok Yoo, Moonjoo Shim, and Dongkyun Kim. 2012. Dynamic duty-cycle scheduling schemes for energy-harvesting wireless sensor networks. *IEEE communications letters* 16, 2 (2012), 202–204.