Short-term Memory Solar Energy Forecasting at University of Illinois

Adele Kuzmiakova "akuzmiakova", Gael Colas "colasg", Alex McKeehan "mckeehan"

December 2017

Abstract: *Climate change and energy crisis have motivated the use and development of solar power generation. Since solar power generation is highly intermittent and dependent on local weather characteristics, we apply a variety of linear and non-linear ensemble machine learning models to predict short-term solar energy output at the University of Illinois campus. Implemented models include weighted linear regression with and without dimension reduction, boosting regression trees, and artificial neural networks with and without vanishing temporal gradient. Additionally, we apply a variety of variable selection techniques, which suggest that air temperature, relative humidity, and dew point are the most informative weather parameters for predicting the solar energy generation at the Illinois power plant. Findings from our paper indicate that deep learning with vanishing time-series gradient (long short-term memory layer; LSTM) is the most promising technique for solar predictions based on temporal weather characteristics. The LSTM layer reduces both median training and test error by an order of magnitude compared to the remaining algorithms. This conclusion confirms the presence of short-term temporal dependence in both weather and solar power magnitudes.*

1 Introduction

Climate change and energy crisis have motivated the use and development of sustainable energy sources, with solar energy being identified as one of the most abundant and promising candidates for bulk power generation. Yet, an inherent characteristic common to all renewable energy sources is that power generation is fully dependent on weather and meteorological parameters, and therefore the solar power output cannot be fully controlled or planned for in advance. To ensure secure integration of photovoltaic (PV) systems into the smart grid, accurate PV forecasting is a critical element of energy management systems. If accurate PV forecasting is lacking, any unexpected fluctuations in solar energy capacity may have significant impacts on the daily operations and physical health of the entire grid and may negatively affect the quality of life of the energy consumers.

As a response to public demand, research in forecasting solar energy has enjoyed a significant degree of interest during the last decade [1, 2, 7, 10, 11]. Current predictive algorithms can be classified into 3 categories: 1) machine learning, 2) physical, and 3) hybrid techniques, which rely on a combination of machine learning algorithms coupled with physical inputs [1, 7]. The machine algorithm studied in this paper fall into the third category and include several linear and non-linear ensemble predictive methods. As a result, the goal of this paper is to implement machine learning algorithms to predict the solar energy output on a university campus at University of Illinois based on its weather and temporal parameters. Key research questions this paper answers are:

1. What weather features are the most influential for solar energy prediction at University of Illinois?
2. Which ML algorithms are the most efficient for predicting weather-driven solar energy output?

To answer the questions, the work presented is structured as follows: Section 2 introduces machine learning algorithms and describes the solar power database and weather parameters used in this study. In Section 3 we describe the algorithmic results, including variable selection results, and linear and non-linear ensemble results. Finally, Section 4 closes off with conclusions and possible hypothesis for alternative directions to approach this problem.

2 Dataset and Features

The solar energy output necessary to power the campus of the University of Illinois in Urbana-Champaign was obtained from publicly-available repository [3]. The weather data for Urbana Champaign, Illinois, was obtained using the methodology detailed by National Oceanographic and Atmospheric Administration [4, 9]. Our Python scripts extract a series of 8 weather parameters, summarized in Table 1. We pre-processed these to obtain the numerical values for each feature and time-averaged them to obtain a consistent hourly resolution. In order to obtain a robust estimate of the weather features in Urbana Champaign, we spatially-averaged them over the 3 closest weather stations using the barycenter formula.

Weather Features	**Unit**	**Weather Features**	**Unit**
Cloud Coverage	% range	Relative Humidity	%
Visibility	Miles	Wind Speed	Mph
Temperature	°C	Station Pressure	inchHg
Dew Point	°C	Altimeter	inchHg

Table 1: A summary of selected meteorological parameters for the Urbana Champaign, Illinois, site.

The time-series for both solar energy and weather parameters starts on February 1, 2016 and ends on October 31, 2017. We use the hourly resolution, ranging from 6AM to 5PM. The total number of examples is 7536. We had to take into account that our examples are not perfectly independent: indeed the solar energy output of an hour of a specific day is obviously correlated with the weather and the energy output of the previous hours of the same day. To remove this dependency, we randomized our dataset and divided the data into 80%-10%-10% partitions for the training, development, and test sets.

3 Methods

In our work we used several machine learning algorithms: weighted linear regression, PCA-based weighted linear regression, boosted regression trees, and neural networks.

In weighted linear regression (WLR), for each new example x, WLR computes a local linear regression: the importance of the training examples in this regression depends on their distance to the given example x. We trained our model on the train set with the cost function:

$$J(\theta)(x) = \frac{1}{2}\sum_{i=1}^{m} w_i(x)(y^{(i)} - \theta^T x^{(i)})^2$$

for the set of weights: $w_i(x) = \exp(-\frac{||x-x^{(i)}||^2}{2\tau^2})$. The importance is parameterized by the bandwidth parameter τ: when τ decreases, the examples close to x are the only one taken into account. To find the optimal value of the bandwidth parameter τ, we ran hold-out cross validation over a wide range of bandwidth parameters, from 2 to 1000. After we narrowed it down progressively, we found that the optimal value was: $\tau = 4$ for WLR. Table 1 above reveals possible physical correlations between weather features, for instance between dew point, temperature, and humidity (e.g. dew point is when the temperature when vapor water starts to condensate). To remove these correlations while keeping the useful variance in the data, we applied principal component analysis (PCA) to our normalized dataset and performed PCA-based weighted linear regression.

Secondly, Gradient Boosting Machine (GBM) algorithm for regression represents a forward-learning ensemble method based on Freund and Schapire's AdaBoost algorithm [10]. The underlying heuristic utilizes increasingly refined approximations to obtain predictive results. GBM algorithm combines two distinct strengths: 1) regression trees, which generate predictions by recursive binary splits to sub-divide the parameters space and minimize loss function that penalizes wrong positive and false negative predictions and 2) boosting, which represents an adaptive method for combining numerous simpler trees (weak learners) into a parent tree (strong learner). Therefore, sequentially, GBM builds regression trees on all the features of the dataset in a fully distributed way where each tree is designed in parallel. The final GBM model represents an additive regression model in which individual trees are fitted in an iterative stagewise process to recover the information content of main parameters. The 3 hyper-parameters to optimize are: number of trees, learning rate, and tree complexity. The learning rate is the fraction of each tree's contribution to the model and tree complexity is the number of nodes in a tree.

The final predictive learning method implemented was an LSTM (long short-term memory) recurrent neural network. We hypothesized that a recurrent neural network would best be able to capture time-dependent trends in the data since feedback loops enable RNNs to exhibit memorization of temporal behavior. Developing the neural network involved sampling performance based on a wide range of modifiable parameters, including the size and number of hidden layers, types of activation functions, type of optimization and regularization, batch and epoch sizes, and cross-validation methods.

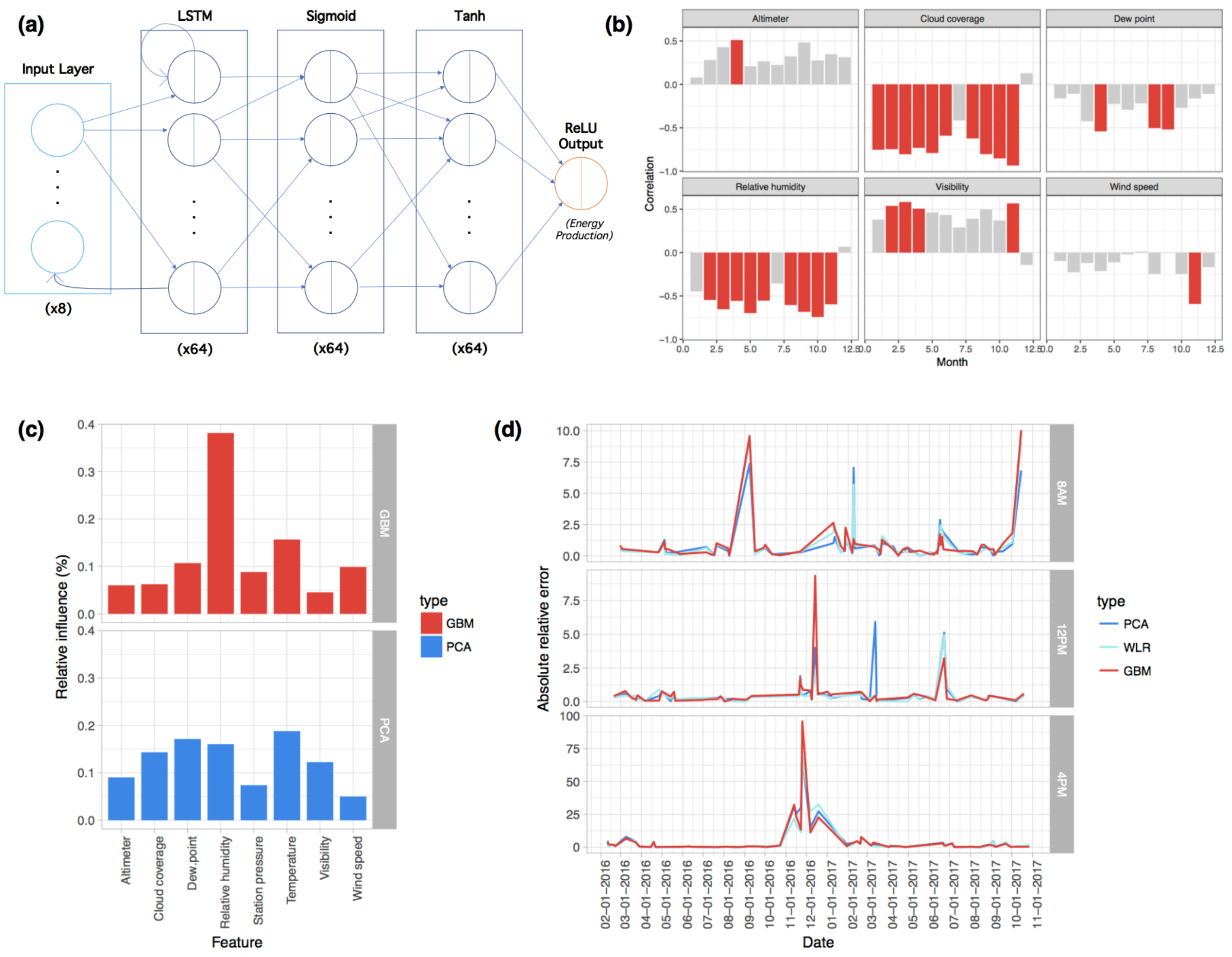

Figure 1: (a) Depiction of flow of hidden layers in optimized neural network. Including an LSTM layer vastly improved performance, while the nonlinear hyperbolic tangent and sigmoid layers exhibited lower errors than standard linear activation functions. (b) Time-based correlations between individual meteorological parameters and actual solar output. Red bars denote values where $|r| > 0.5$, corresponding to strong correlations. (c) Relative influence of meteorological features in the Generalized Boosting Model (GBM; red) and weighted linear regression with PCA (PCA; blue). (d) Absolute relative error from PCA-based regression (PCA; blue), ordinary weighted linear regression (WLR; cyan), and generalized boosting random forest model (GBM; red). Vertical panels correspond to errors at 8AM (morning), 12PM (noon), and 4PM (afternoon).

The optimized neural network comprised three hidden layers (one LSTM) in addition to an input and output layer; Figure 1 a). The introduction of a nonlinear hidden layer and an LSTM layer were each found to greatly increase the accuracy of test predictions. Xavier-He initialization was utilized to select an ideally distributed initial value for the RNN weights. The ‘adam’ optimizer combined the benefits of both RMSProp and AdaGrad in adaptive moment estimation 20% dropout rate to effect regularization [6]. A mean-squared loss function was also used to train the RNN to maintain consistency with the WLR and GBM approaches.

4 Results

4.1 Variable Selection

As a starting point, we performed time-series correlations between all weather parameters and actual energy output in Figure 1 b). The correlations are grouped by month since yearly time scales did not yield meaningful results (i.e. the yearly correlations were found to be on the order of 0). Figure 1 b), we found cloud coverage, relative humidity, visibility, and dew point are the most relevant for the solar output predictions, as evidenced by the absolute values of correlation coefficients exceeding 0.5 (the bars in each sub-panel are highlighted in red if the absolute value of the correlation coefficients r exceed 0.5, which suggests mild-to-strong correlations). Cloud coverage and relative humidity thus have strong correlations for all months but June and December, introducing the hypothesis that these two variables may be the most important attributes for regression and boosting. Figure 1 b) does not contain temperature correlations since, interestingly, their monthly correlations were found to be 0. The anomaly in the wind speed anti-correlation in November is attributed to hurricane-force winds, which were one-time exceptions.

In Figure 1 c), we estimate the relative influence of weather features on predictive performance of PCA-based regression and GBM. In PCA, we calculate the influence of each variable based on their correlations with the largest 5 principal components. In GBM, we calculate the influence based on the number of instances where a particular weather feature is selected for tree splitting, weighted by improvement in the model performance due to each split and averaged across all trees. In both PCA-based regression and GBM, we normalize the contribution of each variable so that their sums add to 100. The higher magnitude suggests a stronger influence on solar response. As Figure 1 c) suggests, one important difference between PCA and GBM's feature selection is that PCA weights all 8 features more uniformly whereas GBM has the propensity to select only a few parameters for tree boosting and weighs them more heavily. For instance in GBM, only 3 attributes exert more than 10% influence on the predictive performance: relative humidity, temperature, and dew point. The GBM variable selection is analogous to backward variable selection in regression, also termed "recursive feature elimination", and works on the principle that non-informative variables are recursively ignored when fitting trees. GBM is characteristic for its ability to identify relevant variables in spite of their mutual interactions, which gives it a predictive advantage over traditional methods, including generalized linear models and clustering [11]. In conclusion, a consensus in variable section from both PCA and GBM is achieved and their feature importance is consistent with their time-series correlation strengths in Figure 1 b).

The most effective approach in terms of RMS training set error was by far the RNN with time-stamp data included as parameters. The most significant gains in model accuracy were found following implementation of an LSTM layer, a nonlinear hyperbolic tangent layer, and the addition of 20% dropout regularization. 10-fold cross validation was used to confirm the stability of predicted results, while a grid search over initializers, batches, epochs, and optimizers was used to confirm Xavier-He initialization, 16, 100, and 'adam' respectively as optimal parameters. The 'sklearn' library was also used to implement boosting of the RNN using AdaBoost and the SAMME.R algorithm. Following these changes, the total mean-squared test set error of the neural network was reduced to 5.439e5, in comparison with a next best test set error of 1.011e6 achieved via standard weighted linear regression – a significant improvement over traditional methods of regression.

4.2 Solar Predictions

To optimize the predictive performance, we chose to select observations ranging from 6AM to 5PM since the nightly energy observations led to a significant increase in variance due to noise. To select the number of principal components for PCA, we evaluated performances from WLR using the range of components (from 1 to 8) and compared their results to results from ordinary WLR with original weather parameters. The minimum median absolute error was achieved when the number of components was 5, giving the basis for the number of PCA components. While processing the dataset with 5 principal components improved significantly the computation time of WLR, the original WLR was found to perform better in the test set.

Figure 1 d) compares the time-series error from 3 regression-based methods: PCA regression (blue), ordinary regression using all 8 weather parameters (cyan), and GBM regression (red). The optimal hyper-parameters in the GBM model were found as follows: tree complexity = 15, learning rate = 0.5, and the number of trees = 35. All 3 regression models give consistent error structure within their specific time intervals across 3 horizontal panels: mornings (8AM), noon (12PM), and afternoon (4PM). We can notice that on average GBM tends to over-predict the response and thus increase the error term (e.g. during September in 2016). Additionally, the highest error terms are generated during winter months (from November to January) due to the very low magnitude of the actual solar output and increased stochastic components. The relatively high degree of stochasticity can be confirmed by comparing error structures between mornings, noon, and afternoons. Here, the error structures vary significantly over the course of one day although it tends to be the highest during winter late afternoons. If we exclude the winter months from the analysis, the median percentage test error is reduced from 41% to 35% for weighted linear regression and from 44% to 29% in the GBM method. Since training parameters may be overly sensitive to low winter values, we also recommend performing the analysis on spring-summer-fall data to validate the parameters or apply higher shrinkage parameter to reduce variance and balance model fit with its predictive performance.

Model	**No. Parameters**	**Training Error**	**Test Error**
PCA-based Weighted Regression	5	0.866*10ˆ6	1.077*10ˆ6
Standard Weighted Regression	8	0.802*10ˆ6	1.011*10ˆ6
Generalized Boosting Method (Random Forest)	3	0.856*10ˆ6	1.240*10ˆ6
Neural Networks (without temporal data)	*	1.160*10ˆ6	1.183*10ˆ6
Neural Networks (with temporal data)	*	5.378*10ˆ5	5.439*10ˆ5

Table 2: Median training and test errors from different machine learning algorithms. *Parameters optimized in neural networks include batch size, no.of epochs, no. of hidden layers, type of hidden layers, no. of neurons, dropout rates, type of optimization, learning rate.

5 Conclusions

In this work, we have performed a study of linear and nonlinear ML algorithms in the context of solar energy forecasting using local weather information. Using dimension reduction based regression and GBM, we determined the most influential features for predicting solar output are temperature, cloud coverage and relative humidity. Comparing all models in Table 1, we noticed that neural network architecture with the LSTM layer reduced both training and test error by an order of magnitude. The conclusion potentially opens up another avenue in this work where a new deep learning algorithm is based on the selection of historical days having similar weather features to the forecasted days. Finding these "days of interests", which carry the most information, could be a supervised learning problem itself. Finally, another research direction we have not explored in this paper is taking advantage of short-term weather dependence instead of randomizing training, development, and test sets. For instance, weather conditions yesterday would dictate weather conditions today to some extent (e.g. we do not expect to encounter a significant temperature gradient overnight). As a result, instead of randomizing our examples, we could have considered them sequentially and attempted to predict the solar output for next day ahead from its immediately preceding one or two neighbors. An example may be a vanishing LSTM gradient in neural networks applied to non-randomized data [13].