Automated Machine Learning for Short-term Electric Load Forecasting

Can Wang, Steffen Limmer, Mitra Baratchi, Thomas Bäck, Holger Hoos, Markus Olhofer

2019

Preprint:

This is an accepted article published in IEEE Symposium Series on Computational Intelligence (SSCI) 2019. The final authenticated version is available online at: https://doi.org/[DOI not available]

Automated Machine Learning for Short-term Electric Load Forecasting

Can Wang
Leiden University
Leiden, the Netherlands
c.wang@liacs.leidenuniv.nl

Steffen Limmer
Honda Research Institute EU
Offenbach am Main, Germany
steffen.limmer@honda-ri.de

Mitra Baratchi
Leiden University
Leiden, the Netherlands
m.baratchi@liacs.leidenuniv.nl

Thomas Bäck
Leiden University
Leiden, the Netherlands
t.h.w.baeck@liacs.leidenuniv.nl

Holger H. Hoos
Leiden University
Leiden, the Netherlands
h.h.hoos@liacs.leidenuniv.nl

Markus Olhofer
Honda Research Institute EU
Offenbach am Main, Germany
markus.olhofer@honda-ri.de

***Abstract*—From detecting skin cancer to translating languages and to forecasting electricity consumption, machine learning is enabling advanced capabilities of computer systems across a broad range of important real-world applications. In this work, we present machine learning models for forecasting the consumption of electricity. Short-term electric load forecasting has been a fundamental concern in power operation systems for over a century. Energy load forecasting is of even greater importance, due to applications in the planning of demand side management, smart electric vehicles and other smart grid technologies. We use two state-of-the-art automated machine learning systems (auto-sklearn and TPOT), which automate model selection and hyperparameter optimization, to achieve maximum prediction accuracy, and compare their performance for the task of load prediction using two benchmark problems. These benchmarks are derived from real world load consumption tasks, namely household consumption from the UCI data repository and consumption data from an industrial office building. Our experimental results indicate great potential for improving the accuracy of energy consumption prediction by using automated machine learning approaches.**

***Keywords*—Machine Learning; Automated Machine Learning; Short-term Load Forecasting; Energy Management.**

I. Introduction

Electricity load forecasting is an important problem for the electric power industry. Many big utility companies have their own load forecasting systems [1]. Load forecasts are needed by other business entities as well, such as banks, insurance companies and stock trading firms.

Due to its strong practical relevance and the challenges involved, load forecasting is an active field of current research. Many approaches to load forecasting can be found in the literature, with a focus on accuracy and efficiency of forecasting models [2], [3], [4].

The primary product of the power industry is electricity, which is moved through the grid to end users. There are two features that make the electricity supply chain very different from other supply chains, e.g., in manufacturing and retail. First, electricity is transmitted very fast, by electrons travelling at the speed of light. Second, since electricity storage in batteries is uneconomical when people only want to transmit energy [5], there is not yet a practical and economical solution for bulk storage of electricity. Therefore, the supply and the demand has to be balanced in real-time.

To plan and operate a system for achieving this balance, we have to understand when, where, and how much electricity is needed throughout the system. The transition to sustainable sources of electrical power critically depends on the management of the highly volatile supply and demand. Without accurate forecasts, end users may experience increasing rates, brownouts or even blackouts. Therefore, the ability to forecast electricity load is crucial to the power industry. Likewise, automating the analysis and optimization of dynamic energy data, as well as measuring decentralized energy production, consumption, storage, and charging in a smart grid environment will contribute to reaching the goal of reduction in realizing a reduction in CO2 emissions of up to 40% by the year 2030 in the EU.

In addition to the energy consumption itself, a number of further data sources can be relevant for load forecasting. For instance, weather data can be relevant, as it can help in estimating energy consumption resulting from heating and cooling. Since consumption patterns change over time, information about work schedules, holidays and special events is also important.

Compared to classic demand or sales data, electricity demand data often comes at substantially higher temporal resolution. Much current research focuses on load forecasting at temporal resolutions of 5 or 15 minutes [1].

Machine learning for short-term electric load forecasting has been studied in prior work [2], [4], [6]. A machine learning pipeline includes tasks such as data cleaning and preprocessing, feature construction, model selection, hyperparameter optimization, and postprocessing of machine learning models. As these tasks all require complicated decision making, the success of machine learning applications crucially relies

Draft submitted to IEEE SSCI 2019

on human machine learning experts, who are familiar with this pipeline, select appropriate machine learning models and set their hyperparameters. The demand for machine learning functionality is growing quite rapidly, and successful machine learning applications can be found in an increasing number of sectors. Since end users in application domains are normally not machine learning experts, there is an urgent need for suitable support in terms of tools that are easy to use.

Automated Machine Learning (AutoML) provides methods and processes that make machine learning accessible to non-machine learning experts. AutoML chooses which machine learning model to use on a given dataset, whether and how to preprocess the dataset, and how to set all hyperparameters at any time for any given dataset without requiring human intervention.

In this work, we apply AutoML for the first time to short-term electric load forecasting tasks. From a data science perspective, the biggest challenge is to develop advanced pipelines of algorithms covering a numerical dynamic data application (real-time, optimized decision making). This pipeline would enable the development of a highly advanced level of decision making, which is impossible to attain for human experts on their own. Thereby, AutoML contributes to the human capital skills in the field of short-term electric load forecasting. Specifically, in this work, we use two state-of-the-art AutoML systems, auto-sklearn [7] and TPOT [8], [9], to forecast the electric load consumption on two benchmarks.

The remainder of this paper is structured as follows: In Section II, a brief overview of existing work about AutoML applications and load forecasting is given. Section III then describes the AutoML systems we used, auto-sklearn and TPOT, in more detail. Section IV, V presents our experimental approach, results and discussion, and finally, some conclusions are given in Section VI.

II. Related Work

Short-term load forecasting techniques are typically classified into two groups: (i) *statistical techniques* and (ii) *artificial intelligence techniques*. The boundary between these two groups is becoming more and more ambiguous, as a result of multidisciplinary collaborations in the scientific community. In the group of statistical techniques, four types of models widely applied to this problem are multiple linear regression (MLR) models [10], [11], semi-parametric additive models [12], [13], auto regressive moving average (ARMA) models [14], and exponential smoothing models [15]. In the other group, four types of AI techniques previously used for this problem are artificial neural networks (ANNs) [16], fuzzy regression models [17], support vector machines (SVMs) [18] and gradient boosting machines [19].

AutoML has been used in many areas, for example, image classification [20], optical character recognition [21], and prediction of the fuel consumption of ships [22] as well as of biological ecosystem networks [23]. Here, we apply AutoML for the first time to the problem of forecasting electric loads.

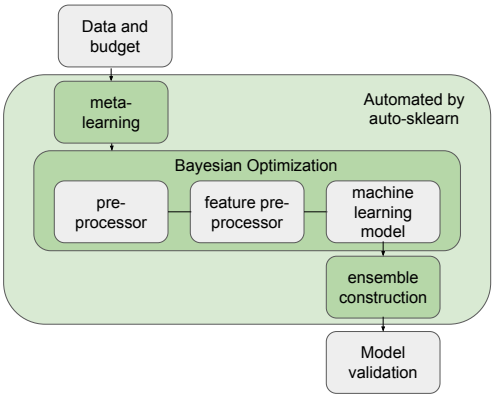

Fig. 1. Auto-sklearn workflow [7].

III. Methods

The AutoML systems we used in our study are auto-sklearn [7] and TPOT [8], [9]. These were chosen because they are state-of-the-art, prominent and freely available.

A. Auto-sklearn

Auto-sklearn [7] uses Bayesian optimization [24] on top of SciKit-learn [25] for generating machine learning pipelines. Auto-sklearn uses SMAC (sequential model-based algorithm configuration) [26] as the underlying Bayesian optimization method to automatically optimize the machine learning pipeline, including preprocessing and postprocessing method selection, model selection and hyperparameter optimization. We note that SMAC is a general-purpose automatic algorithm configurator, and thus not limited to be used in the context of AutoML. It uses random forests as surrogate models in combination with an expected improvement criterion for selecting candidate configurations. The surrogate model is updated throughout the sequential optimization process, which tends to be slow at the beginning, but usually shows good performance over time [7].

AutoWEKA [27] is another AutoML system that uses SMAC as the underlying algorithm configurator. Auto-sklearn differs from Auto-WEKA in two respects: Firstly, auto-sklearn offers the option of using a meta-learning step for initializing the Bayesian optimization process. This can result in a considerable boost in efficiency, as meta-learning can quickly suggest configurations of the ML pipeline that are likely to perform well. In the experiments reported in the following, we do not use meta-learning. Secondly, since it is well known that ensemble models usually show better performance than single models [28], [29], auto-sklearn uses an automated ensemble construction step as a post-processing method to leverage multiple ML models found during the Bayesian optimization process. Auto-sklearn uses a greedy procedure called ensemble selection [30], which starts from an empty set and then iteratively adds the model that maximizes ensemble validation performance. An overview of auto-sklearn workflow is provided in Figure 1.

Auto-sklearn supports 15 classification algorithms: adaptive boosting, Bernoulli naïve Bayes, decision trees, extremely randomized trees, Gaussian naïve Bayes, gradient boosting, k-nearest neighbours, linear discriminant analysis, linear support vector machines, kernel support vector machines, multinomial naïve Bayes, passive aggressive, quadratic discriminant analysis, random forests, and linear classifiers. Regression algorithms are also available in auto-sklearn, including XGBoost, gradient boosting, random forests, k-nearest neighbours, adaptive boosting, decision trees, extra trees, and Gaussian processes.

Auto-sklearn supports 7 categories and 14 possible feature pre-processing methods, including feature selection, kernel approximation, matrix decomposition, embedding, feature clustering, polynomial feature expansion, and using a classifier for feature selection.

B. TPOT

The so-called Tree-based Pipeline Optimization Tool (TPOT) [8], [9] is another AutoML system. It uses genetic programming (GP) [31] to choose the most suitable machine learning model and hyperparameter settings for a machine learning problem. Similar to auto-sklearn, it also searches over machine learning methods available in SciKit-learn [25].

In contrast to auto-sklearn, TPOT allows the use of many copies of a given dataset, which means pre-processing methods can work in parallel and their results can be combined later. By default, the pipeline with the highest accuracy is selected as the final solution returned by TPOT. TPOT can also consider both model complexity and accuracy as optimization objectives, using the NSGA-II selection strategy [32] for selecting candidate configurations for each subsequent generation of the genetic programming process.

The flexibility of TPOT allows for creating machine learning pipelines with any combination of components from SciKit-learn. This, however, can result in invalid pipelines, since there are no constraints on the types of components that are combined into a given pipeline. As a result, resources might be wasted for generating and evaluating invalid pipelines [33].

Similar to other AutoML systems, TPOT is designed for supervised learning tasks. For classification tasks, TPOT supports decision trees, random forests, gradient boosting, logistic regression, k-nearest neighbours and support vector machines. For regression tasks, TPOT covers extra trees, gradient boosting, adaptive boosting, decision trees, XGBoost, random forests, linear SVR, decision trees and k-nearest neighbours.

Furthermore, TPOT supports 2 pre-processors for scaling features: 1) by using the sample mean and variance; 2) by using the sample median and inter-quartile range. Polynomial combinations of numerical features are used to generate additional features. TPOT supports decomposition and uses a variant of principal component analysis. For feature selection, TPOT uses recursive feature elimination strategy, which includes 3 strategies: 1) selecting the top k features; 2) selecting the top n percentile of features; 3) discarding the features that do not satisfy a minimum variance threshold.

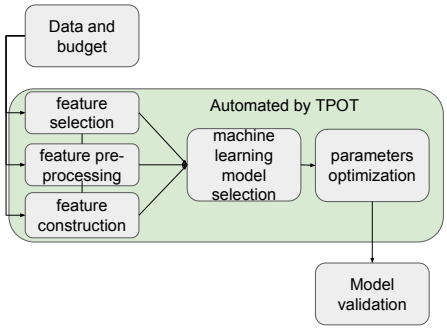

Fig. 2. TPOT workflow [8].

TPOT combines all pipeline components into a tree structure, such that every component is a node in the tree. It begins with one or more copies of the input dataset as the leaves of the tree, then the dataset will be as input for four classes of pipeline components (pre-processor, decomposition, feature selection, or model). When optimizing the pipelines, TPOT evolves these tree structures (i.e., the pipeline components and their hyperparameters) to maximize accuracy.

IV. Experimental Setup and Datasets

In this section, we present our experimental setup and the details of the benchmark datasets that we used.

Experimental setup details: In the experiments, version 0.5.2 of auto-sklearn and version 0.9 of TPOT are used. Each experiment is executed on 8 cores of an Intel Xeon E5-2683 CPU (2.10 GHz) with 10 GB RAM. The time limit for the evaluation of a model is set to 20 minutes for both auto-sklearn and TPOT. Furthermore, MAE is used as performance metric in the optimization. The maximum ensemble size, which is the maximum number of models allowed in the constructed ensemble, is set to the default value of 50 in auto-sklearn. In TPOT, the population size is set to 20, because the default population size of 100 resulted in crashes due to out of memory errors. The mutation rate and crossover rate in TPOT are set to the default values of 0.9 and 0.1, respectively. Since experiments are extremely time-consuming we take the following bootstrapping protocol to create a distribution. We execute every experiment 30 times, then we randomly pick 5 out of 30 and save the best of the 5 results which means the lowest MAE on training data out of these 5 models as one result. This approach is repeated 100 times to create a distribution.

It has to be noted that for the experiments an internal constant WORST_POSSIBLE_RESULT of auto-sklearn is changed from the default value of 1.0 to 2147483647.0. The constant represents the score assigned by SMAC to models that cannot be evaluated due to timeouts or memory errors; it

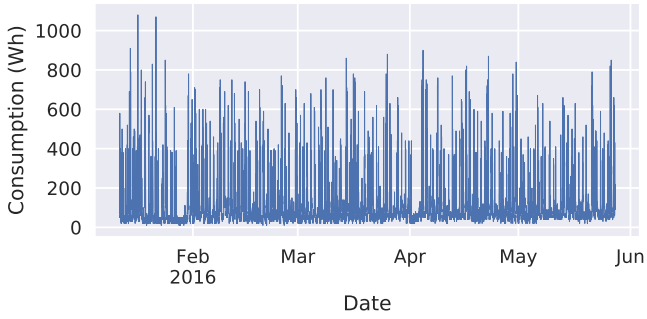

Fig. 3. Energy use of appliances data.

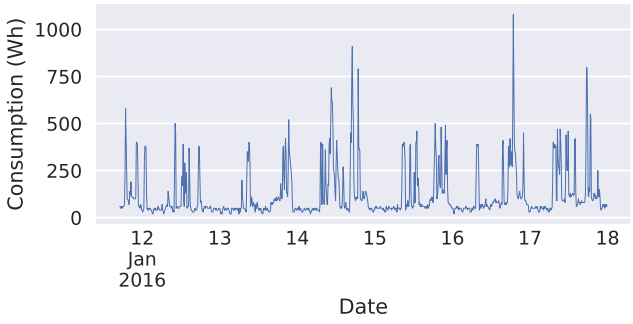

Fig. 4. One week data of energy use of appliances data.

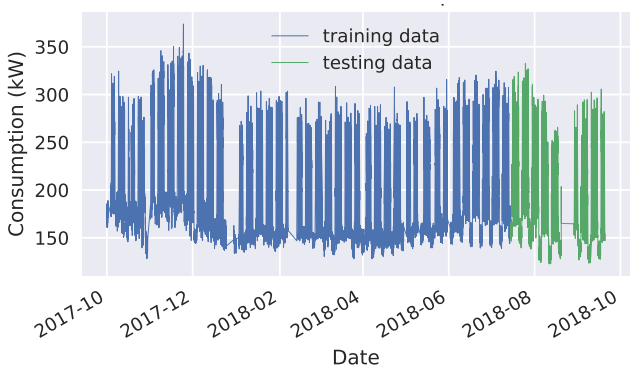

Fig. 5. Honda real world consumption data.

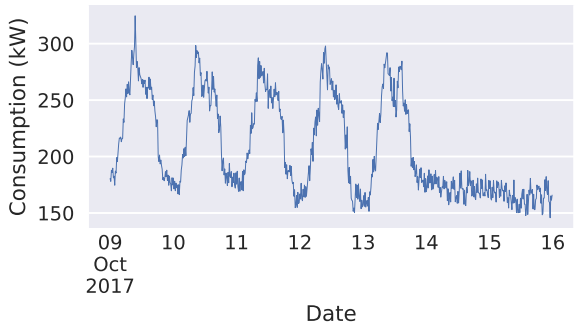

Fig. 6. One week data of Honda real world consumption data.

has to be large in order to steer the automated configuration process away from such models.

Datasets used: The following datasets are used in our experiments:

1) Appliances Consumption Data: The first benchmark dataset we use is a publicly available dataset from the UCI repository [34], [35]. The dataset is made available by the authors of [36], who discuss the use of data-driven models for predicting the energy usage of appliances of a household.

The data set is sampled at 10 minutes for a duration of about 4.5 months. ZigBee wireless sensor network [37] was used to monitor the house temperature and humidity. Each wireless node recorded temperature and humidity conditions every 3.3min, and the temperature and humidity conditions data was averaged over periods of 10 minutes. The energy data was recorded every 10 minutes with m-bus energy meters. Weather information from the airport weather station (Chievres Airport, Belgium) was retrieved from a public data set in Reliable Prognosis (rp5.ru) [36]. It was merged together with experimental data sets by date and time column. Two random variables are included in the data set for testing the regression models. Figures 3 and 4 show the energy consumption of the appliances.

2) Honda Real World Data: The second benchmark dataset is the electricity consumption of an office building of Honda R&D with around 200 employees. The data contains loads and subloads measured with about 40 energy meters, as well as data measured via a weather station. The resolution of the data is 15 minutes and it covers a period from October 2017 to September 2018. Figures 5 and 6 show a part of the measured energy consumption, which we want to forecast in the experiments. We use data from 01-10-2017 to 14-07-2018 as training set and data from 15-07-2018 to 19-09-2018 as testing set.

V. Results

In this section, we present our experimental results. We have two primary goals in performing these experiments: Firstly, we would like to compare the performance of AutoML systems against manually configured models previously proposed for energy consumption forecasting. Secondly, we would like to compare the performance of two currently available AutoML systems for this purpose, namely auto-sklearn and TPOT. For the first goal, we reproduce results previously obtained on two available datasets and compare these results with those obtained by auto-sklearn and TPOT. For the second goal, we compare the performance of auto-sklearn and TPOT under different conditions and using different resources, namely different validation techniques and different training time. Performance of the resulting models is compared using the

TABLE I
EXPERIMENTAL RESULTS ON APPLIANCE CONSUMPTION DATASET.

Model	RMSE	MAE	MAPE%
LM [36]	93.18	51.97	59.93
SVM Radial [36]	70.74	31.36	29.76
GBM [36]	66.65	35.22	38.29
RF [36]	68.48	31.85	31.39
auto-sklearn1	67.80 ± 0.70	29.22 ± 0.48	26.42 ± 0.61
auto-sklearn2	67.01 ± 0.39	28.55 ± 0.28	25.14 ± 0.43
auto-sklearn3	68.99 ± 1.00	29.59 ± 0.45	26.30 ± 0.52
auto-sklearn5	65.37 ± 0.45	27.93 ± 0.45	24.57 ± 0.77
TPOT1	65.12 ± 1.88	28.11 ± 0.77	25.68 ± 0.79
TPOT2	65.70 ± 1.08	27.97 ± 0.70	25.26 ± 0.88
TPOT3	64.89 ± 0.92	27.31 ± 0.50	25.16 ± 0.65
TPOT5	64.23 ± 0.51	27.29 ± 0.17	24.58 ± 0.46

mean absolution error (MAE) metric as this is a metric often used in load forecasting problems. The same metric is also used as the optimization target for AutoML. This means that we try to automatically find the model or ML pipeline that can reach the lowest MAE. We also compare root mean square error (RMSE) and mean absolute percentage error (MAPE).

A. Comparing AutoML Against Manually Configured Models

Appliance consumption data benchmark: In order to compare the performance of AutoML systems against manually configured models we reproduce the experiments presented in [36]. The task to be achieved in this benchmark scenario is to forecast the energy consumption of a house. The authors of [36] have tried all the possible combinations of models and hyperparameters to find the best configuration for a number of model types, including linear regression, support vector machine with radial kernel, random forests and gradient boosting machines. During the training phase, the model performance was evaluated on the validation data sets from a 10 fold cross validation approach.

The best model (SVM Radial) resulted in 31.36 MAE and 29.76 % MAPE on the test data set.

In Table I, we present the results from [36] for LM (linear regression), SVM Radial (support vector machine with radial kernel), GBM (gradient boosting machines) and RF (random forest) and our results obtained with auto-sklearn and TPOT.

Auto-sklearn1, 2, 3, 5 and TPOT1, 2, 3, 5 each represent the final results obtained by auto-sklearn and TPOT with time budgets ranging from 1 to 5 hours. We used the default validation technique setting of auto-sklearn (hold-out) for both auto-sklearn and TPOT in our experiments. We used exactly the same training and testing data as in [36], which was made available by the authors [35]. We use the bootstrapping protocol to create distributions as mentioned in Section IV, and the table provides the mean and standard deviation result values.

Looking at Table I, we observe that auto-sklearn and TPOT both work well on this dataset and beat the baseline. The AutoML methods are able to find an accurate model in one hour which beats the baseline regarding MAE and MAPE. Auto-sklearn achieves 29.22 MAE and 26.42 % MAPE and TPOT results in 28.11 MAE and 25.68 % MAPE on the test data set, which is better than the baseline. And the tendency observed is as follows: giving more time to AutoML techniques, they produce better models. After 5 hours training, auto-sklearn and TPOT both show better performance than the baseline in all the error metrics, i.e., auto-sklearn achieves 65.37 RMSE, 27.93 MAE and 24.57 % MAPE, TPOT achieves 64.23 RMSE, 27.29 MAE and 24.58 % MAPE.

TABLE II
EXPERIMENTAL RESULTS ON HONDA REAL WORLD DATASET.

Model	RMSE	MAE	MAPE%
XGBoost+PA-1	11.40 ± 0.02	8.45 ± 0.01	4.35 ± 0.01
auto-sklearn1	14.35 ± 0.05	11.13 ± 0.03	5.33 ± 0.01
auto-sklearn2	16.29 ± 1.26	12.42 ± 0.96	5.68 ± 0.35
auto-sklearn3	16.79 ± 2.02	12.78 ± 1.56	5.84 ± 0.59
auto-sklearn5	14.52 ± 1.20	11.29 ± 1.08	5.40 ± 0.45
TPOT1	14.47 ± 0.41	11.13 ± 0.49	5.29 ± 0.29
TPOT2	14.32 ± 0.41	11.05 ± 0.33	5.30 ± 0.17
TPOT3	13.81 ± 0.47	10.56 ± 0.47	5.03 ± 0.27
TPOT5	13.73 ± 0.38	10.48 ± 0.31	4.99 ± 0.16

Honda real world dataset benchmark: The task to be achieved in this benchmark experiment is the daily forecast (meaning the forecast of the 96 values of the next day) of the sum of the loads measured by two sensors, which corresponds to about half of the total load. As inputs, historical data of all energy meters can be used in addition to weather data of the day to be forecasted. This task was investigated in a 6 months project in 2018 by a machine learning expert at Honda Research Institute EU. The best results were obtained with an ensemble of XGBoost and PA-1. This setting serves as baseline in the following experiments. PA-1 is an online learning model, which is not available in auto-sklearn or TPOT. In general, current AutoML techniques do not include any online learning model yet. At Honda R&D Europe, power consumption differs markedly between weekdays and weekends. To make a distinction between the days of week, the values 0 and 1, representing weekdays and weekends, respectively, are used as input to the model. We optimize a single pipeline for both weekdays and weekends. We use the load and two subloads from the previous day, the time step within the day to predict, the temperature at this time step, and the weekday flag as input data for the pipeline, and the load at the time step to predict as target variable.

In this experiment, we use hold-out (training:testing = 67:33) as validation technique. We use the bootstrapping protocol to create distributions, and Table II again provides the mean and standard deviations of the performance metrics obtained. We observe from the results shown in Table II that neither TPOT nor auto-sklearn is able to beat the baseline for this dataset within 5 hours. PA-1 is an online learning method that not available in auto-sklearn or TPOT. The baseline is from a six months project, while for AutoML, it only takes

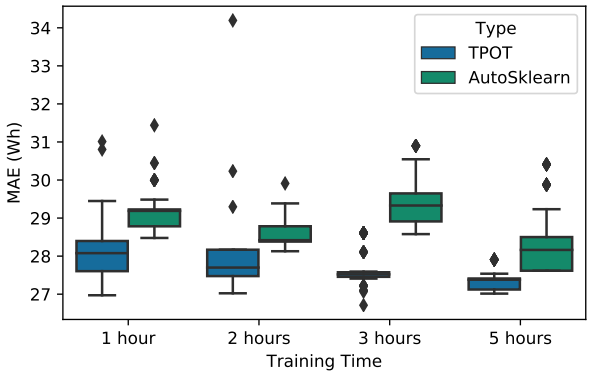

Fig. 7. Testing error on energy use appliance consumption data with different training time.

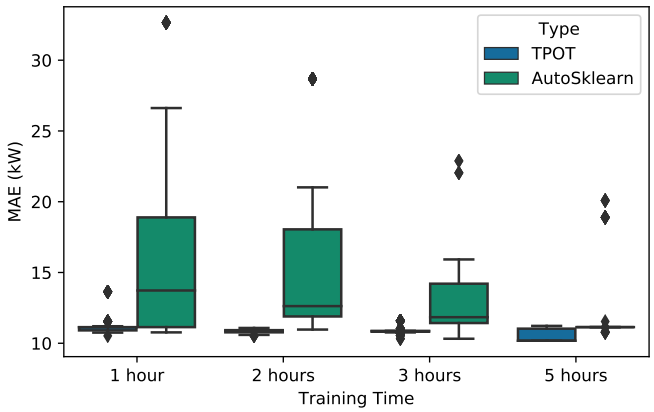

Fig. 8. Testing error on Honda real world data with different training time.

a few hours to find a good model that comes close to the prediction accuracy of an expert-designed forecasting system.

B. Comparing auto-sklearn and TPOT

In this section, we continue experiments by comparing auto-sklearn and TPOT under different conditions and using different resources. First, we compare the error achieved by auto-sklearn and TPOT for different settings of the training time. Then we will compare the error achieved by auto-sklearn and TPOT using different validation techniques. The reason why these factors are relevant are as follows: 1) Longer training time may allow AutoML to explore more of the configuration space. Investigating this, we want to see how fast AutoML finds a good enough configuration. 2) An open problem in AutoML is overfitting [38]. To explore if this problem exists, we compare the error of hold-out versus cross validation and investigate if this choice has any influence on the testing error.

1) Comparison of error metrics optimized by auto-sklearn and TPOT with different training time: In this experiment, we optimize ML pipelines with different time limits from 1 hour to 5 hours and compare the results. The results are presented in Figures 7 and 8. In these figures we compare the testing error on energy use appliance consumption data and Honda real world data, respectively. The results present performance acquired with different training time. In these experiments, we use hold-out (training:testing = 67:33) as validation technique for both auto-sklearn and TPOT. In case of the appliance consumption data as well as the Honda real world data, TPOT generally finds better models (w.r.t. MAE) than auto-sklearn. It is also observed that the performance of TPOT improves when the time budget is increased, while this trend does not generally hold for auto-sklearn.

2) Comparison of error metrics achieved by auto-sklearn and TPOT using different validation technique: In this section, we compare the performance of auto-sklearn and TPOT using two different validation technique. Both hold-out and cross validation can be used for this purpose. The results are

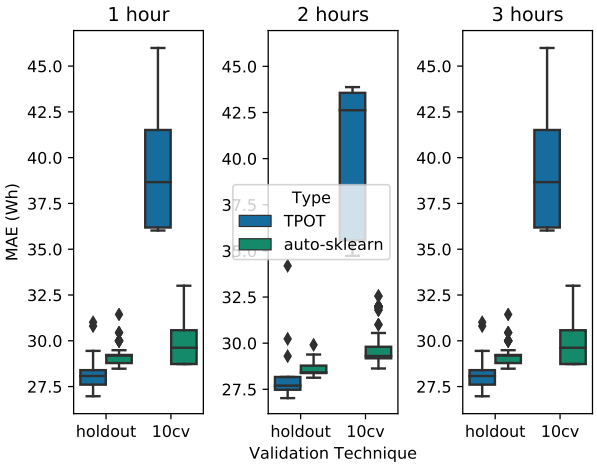

Fig. 9. Testing error with hold-out VS 10 fold cross validation on energy use of appliance consumption data.

presented in Figures 9 and 10. Since training time can also be variable, we have done experiments using different amounts of training time, ranging from 1 hour to 3 hours. Each graph compares results achieved by cross validation versus hold-out for a given dataset and a given amount of training time.

In Figure 9, we compare the testing error on energy use appliance consumption data with different training time, using hold-out (training:testing = 67:33) and 10-fold cross validation as validation technique. We find that TPOT performs very different from auto-sklearn when using a different validation technique. When we use hold-out as validation technique, TPOT always yields relatively good and stable results and a lower error than auto-sklearn. However, when we use 10 fold cross validation as validation technique, the mean as well as standard deviation obtained by TPOT are significantly larger than with hold-out.

In Figure 10, we compare testing error on Honda real world data with different training time and different validation technique. And again, the validation technique hold-out

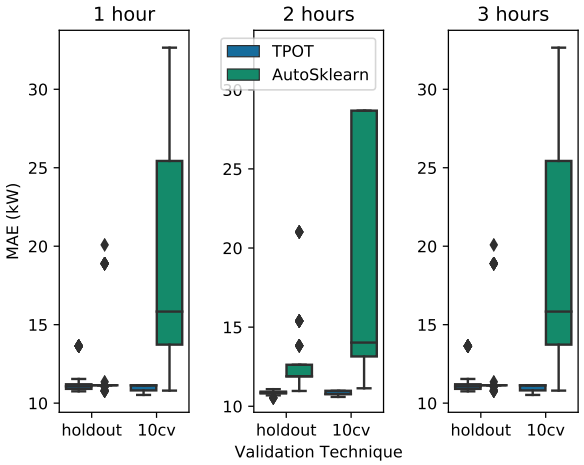

Fig. 10. Testing error with hold-out VS 10 fold cross validation on Honda real world data.

(training:testing = 67:33) and 10 fold cross validation are compared for the different training times. As can be seen in the figure, TPOT yields a lower error than auto-sklearn. In most of the settings, using hold-out as validation technique shows good results for TPOT, while for auto-sklearn we do not see this tendency. The default validation technique of TPOT is cross validation, by changing this into hold-out we can get better or equally good results. We also found with the same computational resources that 10 fold cross validation shows higher variance than hold-out, which seems to result in smaller variances, thus more stable results.

VI. Conclusions

In this work, we investigated the use of AutoML methods for performing load forecasting tasks on two real world datasets. We compared the results achieved by two freely available, state-of-the-art AutoML systems, auto-sklearn with default parameter settings, and TPOT with a minor change, against manually selected models proposed before for the same problem. We found that, using the same computational resources, hold-out validation within the AutoML process tends to produce more stable results than cross-validation. Both AutoML methods show an advantage over the baseline results [36] for the appliance consumption dataset. For the Honda real world data set, AutoML also provides promising results, achieving a performance close to a sophisticated, expert-designed forecasting system. The AutoML systems used in our study are open source and extensible.

Overall, our results clearly indicate that AutoML can substantially reduce the effort involved in building good predictive models for energy consumption forecasting tasks, while also achieving improved accuracy. Therefore, the use of AutoML provides an attractive way for power utilities to improve their load forecasting frameworks. In future work, we plan to extend the use of AutoML systems in this domain to multi-output-models and to online learning approaches.

Acknowledgment

This work is part of the research programme *C2DHorizontal Data Science for Evolving Content*, under the DACCOM-PLI project (project number 628.011.002), which is partly financed by the Netherlands Organisation for Scientific Research (NWO).

References

[1] T. Hong and S. Fan, "Probabilistic electric load forecasting: A tutorial review," *International Journal of Forecasting*, vol. 32, no. 3, pp. 914–938, 2016. [Online]. Available: https://EconPapers.repec.org/RePEc:eee:intfor:v:32:y:2016:i:3:p:914-938

[2] A. K. Srivastava, A. S. Pandey, and D. Singh, "Short-term load forecasting methods: A review," in *2016 International Conference on Emerging Trends in Electrical Electronics Sustainable Energy Systems (ICETEESES)*, March 2016, pp. 130–138.

[3] T. Hong and S. Fan, "Probabilistic electric load forecasting: A tutorial review," *International Journal of Forecasting*, vol. 32, no. 3, pp. 914–938, 2016. [Online]. Available: https://EconPapers.repec.org/RePEc:eee:intfor:v:32:y:2016:i:3:p:914-938

[4] A. Baliyan, K. Gaurav, and S. K. Mishra, "A review of short term load forecasting using artificial neural network models," *Procedia Computer Science*, vol. 48, pp. 121 – 125, 2015, international Conference on Computer, Communication and Convergence (ICCC 2015). [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1877050915006699

[5] I. Staffell and M. Rustomji, "Maximising the value of electricity storage," *Journal of Energy Storage*, vol. 8, pp. 212 – 225, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2352152X1630113X

[6] H. K. Alfares and M. Nazeeruddin, "Electric load forecasting: Literature survey and classification of methods," *Int. J. Systems Science*, vol. 33, pp. 23–34, 2002.

[7] M. Blum, M. Feurer, A. Klein, J. Springenberg, F. Hutter, and K. Eggensperger, "Efficient and robust automated machine learning," in *NIPS*, 2015, Publication, pp. 2962–2970. [Online]. Available: http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning

[8] R. S. Olson, N. Bartley, R. J. Urbanowicz, and J. H. Moore, "Evaluation of a tree-based pipeline optimization tool for automating data science," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, ser. GECCO '16. New York, NY, USA: ACM, 2016, pp. 485–492. [Online]. Available: http://doi.acm.org/10.1145/2908812.2908918

[9] R. S. Olson, R. J. Urbanowicz, P. C. Andrews, N. A. Lavender, L. C. Kidd, and J. H. Moore, *Applications of Evolutionary Computation: 19th European Conference, EvoApplications 2016, Porto, Portugal, March 30 – April 1, 2016, Proceedings, Part I*. Springer International Publishing, 2016, ch. Automating Biomedical Data Science Through Tree-Based Pipeline Optimization, pp. 123–137.

[10] A. Bracale, G. Carpinelli, P. D. Falco, and T. Hong, "Short-term industrial load forecasting: A case study in an italian factory," in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, Sept 2017, pp. 1–6.

[11] P. Wang, B. Liu, and T. Hong, "Electric load forecasting with recency effect: A big data approach," *International Journal of Forecasting*, vol. 32, no. 3, pp. 585–597, 2016. [Online]. Available: https://EconPapers.repec.org/RePEc:eee:intfor:v:32:y:2016:i:3:p:585-597

[12] R. Nedellec, J. Cugliari, and Y. Goude, "Gefcom2012: Electric load forecasting and backcasting with semi-parametric models," *International Journal of Forecasting*, vol. 30, no. 2, pp. 375–381, 2014. [Online]. Available: https://EconPapers.repec.org/RePEc:eee:intfor:v:30:y:2014:i:2:p:375-381

[13] Y. Goude, R. Nedellec, and N. Kong, "Local short and middle term electricity load forecasting with semi-parametric additive models," *IEEE Transactions on Smart Grid*, vol. 5, no. 1, pp. 440–446, Jan 2014.

[14] G. T. Wilson, "Time series analysis: Forecasting and control, 5th edition, by george e. p. box, gwilym m. jenkins, gregory c. reinsel and greta m. ljung, 2015. published by john wiley and sons inc., hoboken, new jersey, pp. 712. isbn: 978-1-118-67502-1," *Journal of Time Series Analysis*, vol. 37, no. 5, pp. 709–711, 2016. [Online]. Available: https://EconPapers.repec.org/RePEc:bla:jtsera:v:37:y:2016:i:5:p:709-711

[15] T. Hong, P. Pinson, and S. Fan, "Global energy forecasting competition 2012," *International Journal of Forecasting*, vol. 30, no. 2, pp. 357 – 363, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169207013000745

[16] P. Chen, S. Liu, C. Shi, B. Hooi, B. Wang, and X. Cheng, "Neucast: Seasonal neural forecast of power grid time series," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden.*, 2018, pp. 3315–3321. [Online]. Available: https://doi.org/10.24963/ijcai.2018/460

[17] T. Hong and P. Wang, "Fuzzy interaction regression for short term load forecasting," Hugo Steinhaus Center, Wroclaw University of Technology, HSC Research Reports HSC/13/14, 2013. [Online]. Available: https://EconPapers.repec.org/RePEc:wuu:wpaper:hsc1314

[18] H. Nie, G. Liu, X. Liu, and Y. Wang, "Hybrid of arima and svms for short-term load forecasting," *Energy Procedia*, vol. 16, pp. 1455 – 1460, 2012, 2012 International Conference on Future Energy, Environment, and Materials. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1876610212002391

[19] J. R. Lloyd, "Gefcom2012 hierarchical load forecasting: Gradient boosting machines and gaussian processes," *International Journal of Forecasting*, vol. 30, no. 2, pp. 369–374, 2014. [Online]. Available: https://EconPapers.repec.org/RePEc:eee:intfor:v:30:y:2014:i:2:p:369-374

[20] B. Zoph, V. Vasudevan, J. Shlens, and Q. Le, "Automl for large scale image classification and object detection," *Available: go. iec. ch/wpai067.[Accessed: 14 September 2018]. Notes Notes Notes Notes International Electrotechnical Commission T*, vol. 41, no. 22, p. 919, 2017.

[21] S. Mahpod and Y. Keller, "Auto-ml deep learning for rashi scripts ocr," *arXiv preprint arXiv:1811.01290*, 2018.

[22] F. Ahlgren and M. Thern, "Auto machine learning for predicting ship fuel consumption," in *ECOS 2018-the 31st International Conference on Efficiency, Cost, Optimization, Simulation and Environmental Impact of Energy Systems*, 2018.

[23] E. Barreiro, C. R. Munteanu, M. Cruz-Monteagudo, A. Pazos, and H. González-Díaz, "Net-net auto machine learning (automl) prediction of complex ecosystems," *Scientific reports*, vol. 8, no. 1, p. 12340, 2018.

[24] S. Theodoridis, *Machine Learning: A Bayesian and Optimization Perspective*, 1st ed. Orlando, FL, USA: Academic Press, Inc., 2015.

[25] L. Buitinck, G. Louppe, M. Blondel, F. Pedregosa, A. Mueller, O. Grisel, V. Niculae, P. Prettenhofer, A. Gramfort, J. Grobler, R. Layton, J. VanderPlas, A. Joly, B. Holt, and G. Varoquaux, "API design for machine learning software: experiences from the scikit-learn project," in *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, 2013, pp. 108–122.

[26] F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Sequential model-based optimization for general algorithm configuration," in *Proceedings of the 5th International Conference on Learning and Intelligent Optimization*, ser. LION'05. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 507–523.

[27] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown, "Auto-weka: Combined selection and hyperparameter optimization of classification algorithms," in *Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '13. New York, NY, USA: ACM, 2013, pp. 847–855. [Online]. Available: http://doi.acm.org/10.1145/2487575.2487629

[28] I. Guyon, A. Saffari, G. Dror, and G. Cawley, "Model selection: Beyond the bayesian/frequentist divide," *J. Mach. Learn. Res.*, vol. 11, pp. 61–87, Mar. 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756006.1756009

[29] A. Lacoste, M. Marchand, F. Laviolette, and H. Larochelle, "Agnostic bayesian learning of ensembles," in *Proceedings of the 31st International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, E. P. Xing and T. Jebara, Eds., vol. 32, no. 1. Bejing, China: PMLR, 22–24 Jun 2014, pp. 611–619. [Online]. Available: http://proceedings.mlr.press/v32/lacoste14.html

[30] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proceedings of the Twenty-first International Conference on Machine Learning*, ser. ICML '04. New York, NY, USA: ACM, 2004, pp. 18–. [Online]. Available: http://doi.acm.org/10.1145/1015330.1015432

[31] J. R. Koza, "Genetic programming as a means for programming computers by natural selection," *Statistics and Computing*, vol. 4, no. 2, pp. 87–112, Jun 1994. [Online]. Available: https://doi.org/10.1007/BF00175355

[32] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002. [Online]. Available: https://doi.org/10.1109/4235.996017

[33] A. G. C. de Sá, W. J. G. S. Pinto, L. O. V. B. Oliveira, and G. L. Pappa, "RECIPE: A grammar-based framework for automatically evolving classification pipelines," in *EuroGP*, ser. Lecture Notes in Computer Science, vol. 10196, 2017, pp. 246–261.

[34] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[35] L. M. Candanedo, "Data driven prediction models of energy use of appliances in a low-energy house," https://github.com/LuisM78/Appliances-energy-prediction-data, 2017.

[36] L. M. Candanedo, V. Feldheim, and D. Deramaix, "Data driven prediction models of energy use of appliances in a low-energy house," *Energy and Buildings*, vol. 140, pp. 81 – 97, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0378778816308970

[37] P. Baronti, P. Pillai, V. W. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and zigbee standards," *Computer communications*, vol. 30, no. 7, pp. 1655–1695, 2007.

[38] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., *Automatic Machine Learning: Methods, Systems, Challenges*. Springer, 2018, in press, available at http://automl.org/book.