Benchmarking Auto-WEKA on a Commodity Machine

João Freitas[1], Nuno Lavado[1] and Jorge Bernardino[1,2]
[1]*Coimbra Polytechnic – ISEC, Rua Pedro Nunes, 3030-199 Coimbra, Portugal*
[2]*CISUC - Centre of Informatics and Systems of University of Coimbra, DEI, Polo 2, 3030-290 Coimbra, Portugal*

Keywords: Data Science, Machine Learning, AutoML, Auto-WEKA, OpenML, Benchmarking.

Abstract: Machine Learning model building is an important and complex task in Data Science but also a good target for automation as recently exploited by AutoML. In general, free and open-source packages offer a joint space of learning algorithms and their respective hyperparameter settings and an optimization method for model search and tuning. In this paper, Auto-WEKA's performance has been tested by running it for short periods of time (5, 15 and 30 minutes) using a commodity machine and suitable datasets with a limited number of observations and features. Benchmarking was performed against the best human-generated solution available in OpenML for each selected dataset. We concluded that increasing the overall time budget available over the previous values didn't improve significantly classifiers' performance.

1 INTRODUCTION

Machine Learning has been applied to solve complex business and research problems and is finding its way into every aspect of computing (Hurwitz and Kirsch, 2018).

Several successful business examples can be listed: customer churn prediction, credit risk modelling, recommendation systems and resource allocation (Almeida et al., 2015). There are also many challenges in research being approached by Machine Learning, for example: disease diagnosis, personalized treatment, and epidemic outbreak prediction.

Data Science traditional pipeline is divided into four major steps: data acquisition, pre-processing, machine learning model building and deployment of ready-to-predict models. Data acquisition aims to identify all relevant and available datasets and depends upon the quality of data governance. Data cleaning, feature engineering and feature selection are the most common pre-processing steps. Machine learning model building involves selecting different algorithms from model families, hyperparameters tuning and training.

Pre-processing is often domain-specific, whereas Machine Learning model building is abstracted away from those complexities, making it a good target for automation (Swearingen et al., 2017).

Recently, a new research area within the Machine Learning community has emerged, called AutoML. It aims the automation of (at least) the Machine Learning model building step.

In the free and open-source area, some solutions are: ATM, that stands for Auto-Tuned Models (Swearingen et al., 2017), Auto-WEKA (Kotthoff et al., 2017; Thornton et al., 2013) and AUTO-SKLEARN (Feurer et al., 2015). Also, a growing number of commercial solutions are being offered. In general, AutoML packages offer a joint space of learning algorithms and their respective hyperparameter settings and an optimization method for model search and tuning. Swearingen *et al.* (2017) provide a comprehensive comparison of ATM, Auto-WEKA and AUTO-SKLEARN.

As far as we know, Auto-WEKA's 2013 version was the first AutoML system. It is built around WEKA, under the push-button interface approach though it also allows command-line interface. No knowledge about the available learning algorithms or their hyperparameters is required. Besides the dataset, the user only needs to provide a memory bound, 1GB by default, and the overall time budget available, 15 minutes by default although the developers recommend running Auto-WEKA with several hours to achieve better results (Kotthoff et al., 2017).

In this study, Auto-WEKA's performance has been tested by running it for short periods of time (5, 15 and 30 minutes) using a commodity machine and suitable datasets with a limited number of

Freitas, J., Lavado, N. and Bernardino, J.
Benchmarking Auto-WEKA on a Commodity Machine.
DOI: 10.5220/0006914801800186
In *Proceedings of the 7th International Conference on Data Science, Technology and Applications (DATA 2018)*, pages 180-186
ISBN: 978-989-758-318-6
Copyright © 2018 by SCITEPRESS – Science and Technology Publications, Lda. All rights reserved

observations and features. It was hypothesized that it would be possible to improve on the so-called control classifier's (Auto-WEKA's with 5 minutes running time) performance by increasing the overall time budget available. Besides the running time of 15 and 30 minutes, it was also included in this study a gold standard (defined in detailed in sub-section 2.3), theoretically related to huge computation time, for benchmarking purposes.

The rest of this paper is structured as follows. Section 2 describes the methods used, and the results are shown in section 3. Section 4 provides the discussion and in section 5 conclusions and future work are presented.

2 METHODS

OpenML is a collaborative online (www.openml.org) environment for machine learning where users can upload and download datasets, tasks and run their models outside the platform. Task object definition ensures reproducible results and several evaluation measures of each run are publicly available (Vanschoren et al., 2013). OpenML was used both for datasets selection and benchmarking as detailed in the following sections.

Shorts periods of Auto-WEKA's running time were used following the same thoughts of its developers, "to accommodate impatient users" (Kotthoff et al., 2017) and also to confer validity on the 15 minutes time budget default.

2.1 Datasets

Datasets have been selected from the OpenML100 benchmark study (OpenML Core Team, n.d.), a carefully curated machine learning benchmark suite of 100 classification datasets suitable for practical experimentation in commodity machines.

In this paper, the focus is given to datasets that besides satisfying all the OpenML100 requirements are characterized by not having missing values and with a binary target feature. In the end, 37 datasets were available from which 18 were randomly selected. The characteristics of these datasets are described in Table 1.

Table 1: Datasets Properties extracted from OpenML.

Data.ID	Observations	Features	Target classes	Numeric features	Nominal features
2	3196	37	2	0	37
37	768	9	2	8	1
44	4601	58	2	57	1
50	958	10	2	0	10
151	45312	9	2	7	2
312	2407	300	2	294	6
333	556	7	2	0	7
334	601	7	2	0	7
335	554	7	2	0	7
1038	3468	971	2	970	1
1046	15545	6	2	5	1
1049	1458	38	2	37	1
1050	1463	38	2	37	1
1063	522	22	2	21	1
1067	2109	22	2	21	1
1068	1109	22	2	21	1
1461	45211	17	2	7	10
1462	1372	5	2	4	1

SCITEPRESS SCIENCE AND TECHNOLOGY PUBLICATIONS

The OpenML100 requirements are the following:

- The number of observations is between 500 and 100000 to focus on medium-sized datasets, that are not too small for proper training and not too big for practical experimentation;
- The number of features does not exceed 5000 features to keep the runtime of algorithms low;
- The target feature has at least two classes;
- The ratio of the minority class and the majority class is above 0.05 to eliminate highly imbalanced datasets that would obfuscate a clear analysis.

The OpenML100 authors' have also excluded datasets which:

- Cannot be randomized via a 10-fold cross-validation due to grouped samples;
- Have an unknown origin or no clearly defined task;
- Include sparse data (e.g., text mining datasets).

These OpenML100 are a *de facto* standard used for benchmarking purposes with several thousands runs available in the platform.

2.2 Auto-WEKA Setup

The main objective of this benchmark study was to use a commodity machine that anyone could have access to. The two machines used have the following characteristics:

Machine 1:

- 8GB of RAM;
- Intel I5-6500 CPU 3.20 GHz;
- 250 GB SSD.

Machine 2:

- 8GB of RAM;
- Intel Pentium CPU G3260 3.30 GHz;
- 1 TB HDD.

Notice that two machines were only used to accelerate the experiments because instead of one dataset we could run two at the same time, one in each machine. Both machines used Ubuntu 16.04 LTS 64 Bit version as the Operating System.

To find the best model for each dataset the same Auto-Weka configuration was used, as shown in Figure 1, except for the overall time budget available *timeLimit* that assumed different running times: 5, 15, and 30 minutes.

The memory bound *memLimit* was changed to 2GB, the default is 1GB.

There are several metrics to determine the performance of classifiers. These are available under *metric*. The default is *errorRate*, the rate of incorrectly predicted examples in an unseen test dataset. The *metric* used was precision, because contrarily to errorRate metric, this option could be used directly for comparisons with OpenML results. Precision reflects the proportion of the examples which truly belong to a certain class among all those which were classified as belonging to that class (Almeida et al., 2016).

The following command-line call for running Auto-WEKA on the training dataset *mydata.arff* would produce the same results as the selection in Figure 1.

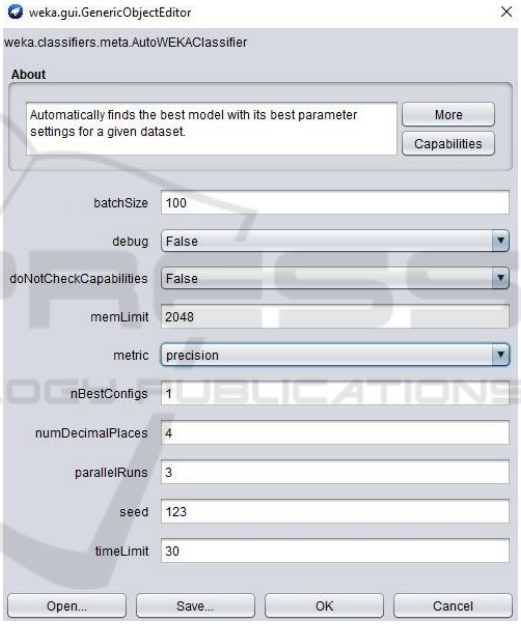

Figure 1: Graphical user interface for Auto-WEKA configuration options.

```
java -cp autoweka.jar
weka.classifiers.meta.AutoWEKAClassifier
-memLimit 2048 -timeLimit 30
-metric precision -t mydata.arff -no-cv
```

Notice that by default WEKA performs and reports stratified cross-validation performance metrics. Setting the flag `-no-cv` forces WEKA to skip it because Auto-WEKA already performs a statistically rigorous evaluation internally (10 fold cross-validation) and doing it would not improve the quality of the result and cause Auto-WEKA to take much longer (Kotthoff et al., 2017).

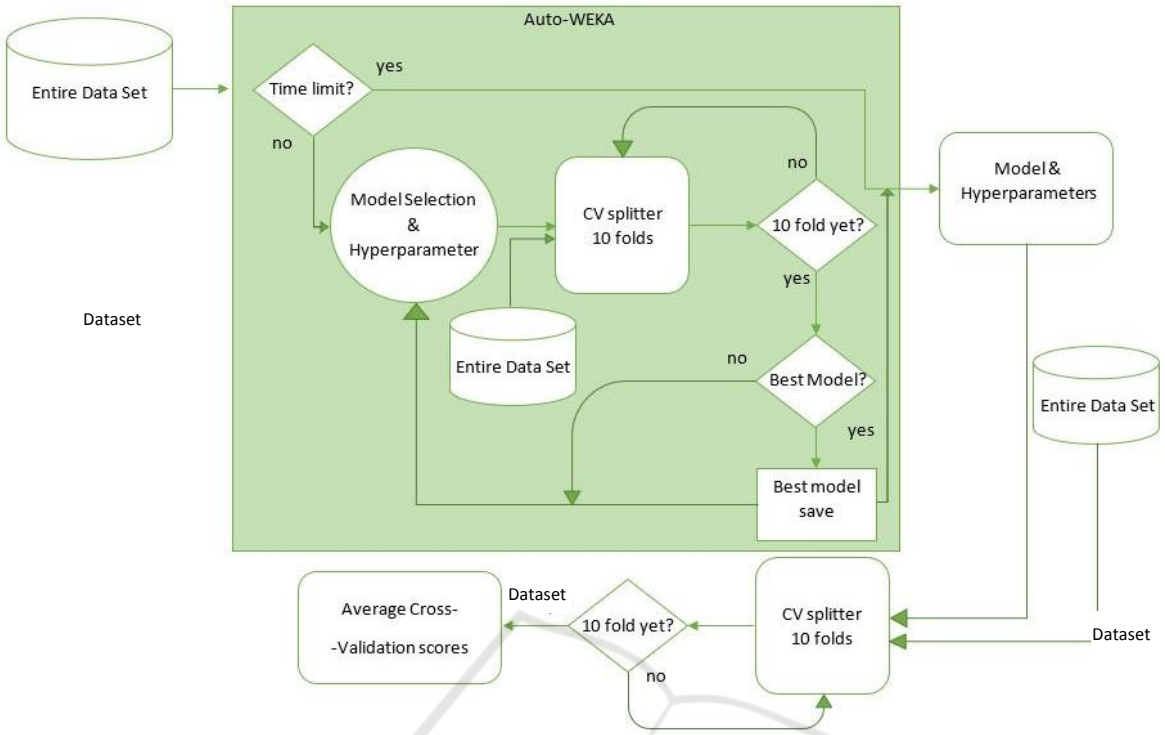

Figure 2: Result extraction through Auto-WEKA.

In order to have access to stratified 10-fold cross-validation performance, after each Auto-WEKA run, the best classifier and, if any, the features/attributes selection were included in a command-line call for running WEKA with a different seed, without the flag `-no-cv`.

Figure 2 illustrates the pipeline followed to extract the results shown in section 3 of this benchmark study. Auto-WEKA uses predictive models to determine which algorithms and hyperparameter configurations are evaluated (Thornton et al., 2013) and for each trains and test it using 10-fold cross-validation on the entire dataset. For each configuration, it compares the precision result with the best model saved, if it is better this model is replaced with the new one. After the set time limit, the best model is exported alongside with the best hyperparameters. An independent CV is done afterwards with the best configuration using the entire dataset to compute the final cross-validation score. Notice that the best configuration used as an input for this last step comprises the optimal hyperparameters and algorithm but not any information about the optimal model parameters.

The command-line below illustrates this procedure, by extracting the relevant information from Figure 3.

```
java -cp weka.jar
weka.classifiers.meta.AttributeSelec
tedClassifier -c 7 -x 10 -s 7 -t
mydata.arff -S ".GreedyStepwise -B -
R" -E ".CfsSubsetEval -M -L " -W
.Bagging -- -P 92 -I 52 -S 1 -W
weka.classifiers.rules.PART -- -M 8.
```

```
Auto-WEKA result:
best classifier:
weka.classifiers.meta.Bagging
arguments: [-P, 92, -I, 52, -S, 1, -W,
weka.classifiers.rules.PART, --, -M, 8]
attribute search:
weka.attributeSelection.GreedyStepwise
attribute search arguments: [-B, -R]
attribute evaluation:
weka.attributeSelection.CfsSubsetEval
attribute evaluation arguments: [-M, -L]
metric: precision
estimated precision: 0.8866995073891626
training time on evaluation dataset:
1.293 seconds
```

Figure 3: Example of Auto-WEKA result.

In the command the flags have the following meaning:

- -c: target class index;
- -x: number of cross-validation folds.;
- -s: seed number.;

- -t: path to dataset;
- -S: attribute search algorithm and parameters;
- -E: attribute evaluation algorithm and parameters;
- -W: Weka classifier and parameters.

The following pre-tests were made to guarantee that Auto-WEKA would generate results without using any cache or temporary files:

- 3 times 30 minutes runs with reset between the runs;
- 3 times 5 minutes runs without reset between runs.

In both cases, it was verified that for the same dataset results were the same so we concluded that Auto-WEKA gives the same results anytime we re-run it.

2.3 Auto-WEKA Benchmarking

The best human-generated solution available in OpenML for each of the 18 selected datasets were access and their precision recorded. These precisions are the top submissions of thousands of runs and thus considered as the gold standard, these values are shown in the last column of Table 2. Also, one can argue, at least from a theoretical point of view, that these are related to huge computation time (all runs summed up) and rely on the best machine learning model building "tool" that's available nowadays, the human Data Scientist.

In order to test the hypothesis that there is a significant improvement on control classifier's performance by increasing the overall time budget available (5min., 15 min., 30 min. and the gold standard) it was used the Friedman test, a non-parametric equivalent of the repeated-measures ANOVA. Also, Bonferroni post-hoc test would be used for comparison of each approach with the control classifier if Friedman test rejects its related null hypothesis. Demsar (2006) provides a comprehensive review of these classical statistical procedures applied to the comparison of machine learning classifiers over multiple datasets.

3 RESULTS

Table 2 shows the 10-fold Cross-Validation precision results for Auto-WEKA with the overall time budget set to 5, 15, and 30 minutes for each dataset. The fourth column shows the gold standard precision as achieved by OpenML best human-generated solutions. Notice that latter precision was also calculated using 10-fold Cross-Validation as defined in the respective OpenML tasks. The last row in Table 2 shows the average rank across all datasets as computed for the Friedman test: for each dataset precisions are ranked starting from the best performing approach, assigning a rank between 1 and 4.

Table 2 suggests that the best performance is achieved by the best human-generated solution available in OpenML, while Auto-WEKA's shows no major average difference when computed with 5, 15 or 30 minutes.

Table 2: 10-fold Cross-Validation precision results for Auto-WEKA and OpenML best human-generated and average ranks.

	Auto-WEKA			Human-Generated
Data.ID	5 min	15 min	30 min	OpenML
3	0,943	0,943	0,943	0,998
37	0,729	0,729	0,729	0,784
44	0,934	0,934	0,940	0,963
50	0,819	0,997	0,819	1,000
151	0,782	0,782	0,782	0,950
312	0,915	0,963	0,962	0,990
333	0,774	0,774	0,774	1,000
334	0,876	0,526	0,499	1,000
335	0,989	0,907	0,987	0,989
1038	0,888	0,888	0,888	0,963
1046	0,946	0,946	0,946	0,963
1049	0,905	0,905	0,905	0,921
1050	0,870	0,856	0,870	0,909
1063	0,824	0,822	0,846	0,862
1067	0,821	0,845	0,845	0,873
1068	0,917	0,919	0,917	0,943
1461	0,864	0,864	0,864	0,902
1462	0,935	0,935	0,935	1,000
Average Rank	**3,111**	**2,889**	**3,000**	**1,000**

According to Friedman's test, there's statistical evidence to support the claim that there's a significant improvement on control classifier's performance by increasing the overall time budget available (Friedman chi-squared=42.04; df=3; p-value<0.00; p-value less than 0.05 means that the difference between the average ranks is significant). However,

Bonferroni post-hoc test confirms that only the best human-generated solution available in OpenML performs significantly better than Auto-WEKA with 5 min running time. This means that there's no evidence arising from this experiment that confirms that increasing the running time from 5 minutes to 15 or 30 minutes will improve the precision of the results.

4 DISCUSSION OF RESULTS

Auto-WEKA's default values, memory bound of 1GB and 15 minutes of overall time budget available, suggests that a classification task running in a commodity machine would achieve a reasonable performance for suitable datasets. Experiments were conducted using 18 datasets with a limited number of observations and features.

The results show that even for such classification tasks, Auto-WEKA with low computing time performs poorly when compared to the best human-generated solution available in OpenML. Also, running times up to 30 min seem to achieve the same performance. Either the selected classification tasks are harder than expected or Auto-WEKA's default values should be revised so that the user is not misled.

However, from another point of view, one can argue that the difference between the best human-generated solution available in OpenML and Auto-WEKA is less than 6 percentage points for 66% of the datasets. That might be a good result for such a little computation time, at least for a baseline precision.

5 CONCLUSIONS AND FUTURE WORK

Auto-WEKA is a powerful tool that allows anyone to fit a model to a dataset, giving, not only the best model for a giving time budget but also an attribute/feature selection approach. It is user-friendly and multi-platform, both must have characteristics for nowadays apps. While looking for a tool that can provide a model without spending too much time, Auto-WEKA may be the solution, despite the precision of the chosen model may not be the best of the best, it can give a solid start.

The results were promising, and one can obviously expect Auto-WEKA to find a better model using High-Performance Computers (HPC) clusters.

As far as we know there are no other studies on the performance on Auto-WEKA in a commodity machine using the default values. Increasing the overall time budget available from 5 to 15 and 30 minutes didn't improve significantly classifiers' performance. Therefore, increasing only by a few minutes doesn't seem to have an impact on Auto-WEKA's performance. Thus, Auto-WEKA's default values (15 minutes and 1GB) might be misleading for the user as there is no evidence that a good classification performance is achieved with such low running times and memory limit.

In this study, only the best human-generated solution available in OpenML, that is related to huge accumulated computation time performed signify-cantly better than Auto-WEKA's 5 minutes run.

There are some questions that we intend to explore in the future. For each dataset how many algorithms and hyperparameter configurations were evaluated in the provided time budget. Which characteristics of the dataset affect Auto-WEKA's performance. What's the effect of increasing running time to more than 30 minutes.

As future work, we plan to run several experiments in parallel using a HPC cluster. We intend to understand how much running time is needed for Auto-WEKA to converge to the best human-generated solution.

ACKNOWLEDGEMENTS

This work was supported by Coimbra Polytechnic – ISEC, Coimbra, Portugal. The authors are also grateful to the reviewers who kindly suggested improvements on the present paper.

REFERENCES

Almeida P., Gruenwald L. and Bernardino J. 2016. Evaluating Open Source Data Mining Tools for Business.In Proceedings of the 5th International Conference on Data Management Technologies and Applications - Volume 1: DATA, ISBN 978-989-758-193-9, pages 87-94. DOI: 10.5220/0005939900870094

Almeida P., and Bernardino J., 2015. A Comprehensive Overview of Open Source Big Data Platforms and Frameworks. International Journal of Big Data, Vol. 2, No. 3, pp. 1-19.

Demsar, J., 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research*, 7: 1-30.

Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F., 2015. Efficient and Robust Automated Machine Learning. In *Advances in Neural*

Information Processing Systems 28-NIPS2015. Curran Associates, Inc., pp. 2962-2970.

Hurwitz, J., Kirsch, D., 2018. *Machine Learning for Dummies, IBM Limited Edition*, John Wiley & Sons, Inc. New Jersey, 1st edition.

Kotthoff, L., Thornton, C., Hoos, H., Hutter, F., Leyton-Brown, K., 2017. Auto-WEKA 2.0: Automatic model selection and hyperparameter optimization in WEKA. *Journal of Machine Learning Research*, 18: 1-5.

OpenML Core Team, n.d. Benchmarking. Retrieved from https://docs.openml.org/benchmark/

Swearingen, T., Drevo, W., Cyphers, B., Cuesta-Infante, A., Ross, A., Veeramachaneni, K., 2017. ATM: A distributed, collaborative, scalable system for automated machine learning. In *2017 IEEE International Conference on Big Data (Big Data)*. Boston, MA, pp.151-162.

Thornton, C., Hutter, F., Hoos, H., Leyton-Brown, K., 2013. Auto-WEKA: Combined Selection and Hyperparameter Optimization of Classification Algorithms. In *Proceedings of the 19th ACM SIGKDD International Conference on KDD*. Chicago, IL, pp.847-855.

Vanschoren, J., Rijn, J., Bischl, B., Torgo, L., 2013. Openml: Networked science in machine learning. *SIGKDD Explorations*, vol. 15, 2, pp. 49-60.

SCITEPRESS
SCIENCE AND TECHNOLOGY PUBLICATIONS