A Tool for Measuring Energy Consumption in Data Stream Mining

Eric Kenzo Taniguchi Onuki[1 [0009−0008−6478−9617]], Andreia Malucelli[1 [0000−0002−0929−1874]], and Jean Paul Barddal[1 [0000−0001−9928−854X]]

Programa de Pós-Graduação em Informática (PPGIa)
Pontifícia Universidade Católica do Paraná (PUCPR), Curitiba, Paraná, Brazil
{eric.kenzo, malu, jean.barddal}@ppgia.pucpr.br

Abstract. Energy consumption reduction is an increasing trend in machine learning given its relevance in socio-ecological importance. Consequently, it is important to quantify how real-time learning algorithms tailored for data streams and edge computing behave in terms of accuracy, processing time, memory usage, and energy consumption. In this work, we bring forward a tool for measuring energy consumption in the Massive Online Analysis (MOA). First, we analyze the energy consumption rates obtained by our tool against a gold-standard hardware solution, thus showing the robustness of our approach. Next, we experimentally analyze classification algorithms under different validation protocols and concept drift and highlight how such classifiers behave under such conditions. Results show that our tools enable the identification of different classifiers' energy consumption. In particular, it allows a better understanding of how energy consumption rates vary in drifting and non-drifting scenarios. Finally, given the insights obtained during experimentation on existing classifiers, we make our tool publicly available to the scientific community so that energy consumption is also accounted for in developing and comparing data stream mining algorithms.

Keywords: Data stream mining · Energy consumption · Green computing.

1 Introduction

Data is consistently being generated, stored, and processed. Several applications provide data generated in large amounts, frequency, and speed. In this paper, we focus on such scenarios the so-called data streams. Data streams are, per definition, potentially unbounded and non-stationary data sequences [10]. Consequently, performing data mining to extract useful insights and patterns from such data requires algorithms that are tailored to such scenarios.

In contrast to traditional machine learning, the data stream mining area has shown concerns with the trade-off between accuracy and computational resources, i.e., processing time and memory consumption [5]. Nonetheless, recent research has shown that we still need more significant steps toward sustainability. For instance, authors in [28] depict that the CO2 emissions of training

neural networks are larger than that of a car in its lifetime. First, there is no clear relationship between processing time, memory consumption, and energy consumption. Even though these components are tied to one another, multi-threading, compiler, and other low-level computational architecture have been shown to impact the entire process, and, consequently, energy consumption cannot be directly estimated from such components [16, 17].

Having both sustainability and the lack of generic tools for quantifying energy consumption in streaming scenarios as motivation, we bring forward a tool for researchers and practitioners to investigate how data stream mining algorithms behave under different streaming settings, e.g., with and without concept drifts, different validation schema, etc. In opposition to previous works [13, 14], our tool is generic because it can be coupled with any classifier and data stream available in the Massive Online Analysis (MOA) framework [6], which is the off-the-shelf solution for implementing and testing streaming methods. We experimentally evaluate our tool against a hardware solution and assess the energy consumption of different classifiers under different streaming settings. Finally, the tool is made publicly available to the scientific community as a byproduct of our research.

This paper is divided as follows. Section 2 describes data stream mining and brings forward the main concepts in energy consumption. Section 3 discusses related works that lie at the intersection of energy consumption and data stream mining. Section 4 describes our tool for energy consumption measurements and describes how it has been combined with the Massive Online Analysis (MOA) [6] framework. Section 5 discusses the analysis conducted to validate our tool and assess different classifiers under different experimental conditions. Finally, Section 6 concludes this work and states envisioned future works.

2 Data Stream Mining and Energy Consumption

Data streams are potentially unbounded data sequences made available over time, which may be non-stationary. Consequently, storing an entire data stream is unfeasible since it is not entirely available at once, and it would not fit in memory [10]. As a result, researchers and practitioners have devoted efforts towards developing efficient algorithms to process and mine data that arrive sequentially over time. Therefore, data stream mining is understood as the investigation of patterns, anomalies, and correlations in streaming data. In particular, in this work, we focus on classification, the most popular task in data stream mining that conveys the prediction of a discrete output given a set of input variables. More formally, we denote a data stream S to provide instances $i^t = (\boldsymbol{x}^t, y^t)$ at timestamps denoted as t. We also denote classification as the task of learning a predictive model $f : \boldsymbol{x} \rightarrow y$, where y is a discrete label in Y. In practice, we expect predictions $\hat{y}$ to be accurate given the ground-truth y values.

One of the main challenges in data stream mining is concept drift [31], which regards changes in the data distribution that may render a classifier obsolete. Formally, a concept $C = \bigcup_{y_i \in Y} \{(P[y_i], P[\boldsymbol{x}|y_i])\}$ is a set of class priors and class-conditional probability density functions [10]. Therefore, a concept drift

is said to occur between two timestamps t_i and t_j if $C^{t_i} \neq C^{t_j}$ [12]. Consequently, classifiers for data streams must be adaptive, which means that f may be adjusted when newly labeled instances are made available.

2.1 Classifiers

Over the years, different approaches have been developed for the classification task in data streams. In practice, these classifiers are variants of traditional classifiers available for batch scenarios.

A popular approach for classification in streaming scenarios is the Incremental Naive Bayes [24]. As its batch counterpart, it assumes that input features are independent. With the arrival of a training instance, all probabilities are updated according. Since probabilities are based on counters, and there is no need to store instances, Naive Bayes has a constant memory consumption and processing time. Nonetheless, it does not present any traits to identify and adapt to concept drifts.

The most common approach for learning from data streams is decision trees. In particular, Hoeffding Trees [9] is the most popular approach as it branches over time when statistically enough data (grace period, $n_{\min}$) and evidence has been gathered, according to the Hoeffding Bound [22]. The Very Fast Decision Tree (VFDT) is a popular implementation of incremental Hoeffding Trees, meaning that it continuously branches as new data becomes available and does not revisit the quality of previously created split nodes. In contrast, the characteristic of revisiting split nodes is observed in Hoeffding Adaptive Trees [4], in which each split node is coupled with an ADWIN drift detector [3]. Whenever a drift is flagged, the corresponding split node is replaced by a leaf node, which can branch again if the Hoeffding inequality is met. Even though Hoeffding Adaptive Trees significantly improve accuracy rates compared to incremental trees, even better results are obtained when creating ensembles of Hoeffding Trees. A state-of-the-art exemplar of ensembles of Hoeffding Trees is the Adaptive Random Forest (ARF) [19], in which Randomized Hoeffding Trees are trained in parallel and coupled with drift detectors to identify and adapt to concept drifts rapidly. ARF adjusts the sampling process with Poisson($\lambda = 6$) so that instances have higher chances of being used during training, thus speeding up the drift adaptation process. In the test step, classifiers' votes are combined using weighted majority voting, i.e., classifiers with higher accuracy have a higher impact on the final prediction. Consequently, ARF is a strong learner that achieves state-of-the-art results in terms of accuracy, yet, at the high expense of computational resources.

2.2 Requirements

Throughout the training and test steps of data stream classification, streaming classifiers must meet certain requirements [5, 6]:

- **Requirement #1**: Process an example at a time, and inspect it only once (at most);

- **Requirement #2**: Use a limited amount of memory;
- **Requirement #3**: Work in a limited amount of time;
- **Requirement #4**: Be ready to predict at any point; and
- **Requirement #5**: Detect and adapt to concept drifts.

This list has been incremented in the works of García-Martín [15, 17, 18], in which energy consumption is highlighted as a relevant aspect in data stream mining since several classifiers have been tailored focusing solely on accuracy and overlooking sustainability. This is one of the main drivers of our work, i.e., to allow researchers and practitioners to quantify the energy consumption of data stream classifiers and determine under which conditions they fail to meet energy sustainability criteria.

3 Related Works

Over the years, different approaches to measuring energy consumption have been proposed. This section highlights approaches for measuring energy consumption in data stream mining and decreasing energy usage. A first significant study is [14], in which authors used PowerAPI [30] to quantify the energy consumption of Hoeffding Trees despite acknowledging that it overlooks RAM consumption. The same authors have changed their approach in [13], in which Jalen (now called JourlarX) [26] has been used to quantify energy consumption of Hoeffding Trees on a function-basis. This allowed the authors to identify bottlenecks in the existing Hoeffding Tree implementation available in MOA.

Finally, authors in [17] and [18] have used Intel's RAPL [7] to quantify the energy consumed by the DRAM and the processor based on accesses to the processor performance counters. Even though Intel's RAPL code is not available, authors disclose that its accuracy has been checked in [21] and that it does not introduce processing overheads. It is also relevant to highlight that the work of [18] introduces a Hoeffding Tree variant in which the grace period ($n_{\min}$) is adjusted so that branching is only attempted according to a user-given threshold. The results showed that the proposed Hoeffding Tree variant has accuracy convergence while approximately 65% less energy consumption rates. On the other hand, the work of [17] introduces a framework to quantify the energy consumption of Hoeffding Tree ensembles while accounting for decision tree learning, drift detectors, and tree replacement.

Regarding all of the works mentioned above, a significant drawback is that energy consumption has been tackled solely for Hoeffding Trees, and no general open-source tool makes energy consumption is easily available for researchers and practitioners. Our proposal is brought forward in the next section, and it circumvents such problems.

4 Proposal

In this section, we detail our tool to quantify the energy consumption of data stream mining algorithms. Our tool is embedded within the Massive Online

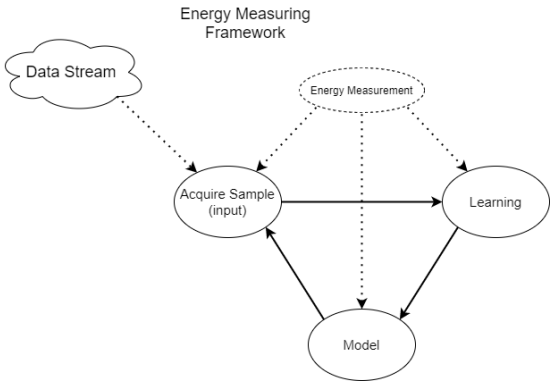

Fig. 1. Overview of the proposed energy measuring framework. The tool measures energy consumption during data acquisition and model training and testing.

Analysis (MOA) framework [6], yet, its rationale can be used in other tools like River [25].

Our tool uses Intel's RAPL [7] and can be seen as a plugin to the Massive Online Analysis (MOA) framework. The general idea of our tool is given in Figure 1, in which we see that RAPL is used to quantify energy consumption in data acquisition and models' training and testing phases. Once an experiment starts, energy measurements are initialized. During data stream processing, arriving instances are used to determine processing and energy readings before and after processing, i.e., testing and training steps. These readings are used to compute energy consumption rates during the experiment. For each of the stages, there are different forms of measuring energy consumption. The framework controls the flow of the data stream model to start the energy measurement right before it begins processing the samples. At each cycle, the measurement is taken and presented in real-time to the user. At the end of the process, a graph showing the instantaneous measurement for each cycle is presented to the user.

Since our tool is based on Intel RAPL, we provide in Figure 2 details on how our plugin interacts with MOA, RAPL, and the Linux Kernel. As new data becomes available for processing, the plugin requests measurements from the Linux Kernel, receiving the response of how much energy has been spent during the testing and training phases. These values are summed and made available whenever the evaluation interface (the so-called evaluation frequency parameter) requests an energy consumption rate. Figure 3 exemplifies the energy consumption rates measured by the proposed tool and how it is reported in MOA alongside other evaluation metrics.

The source code of our proposal and experimentation can be found at `https://github.com/ericonuki/moa-bringing-awareness-green-ict`.

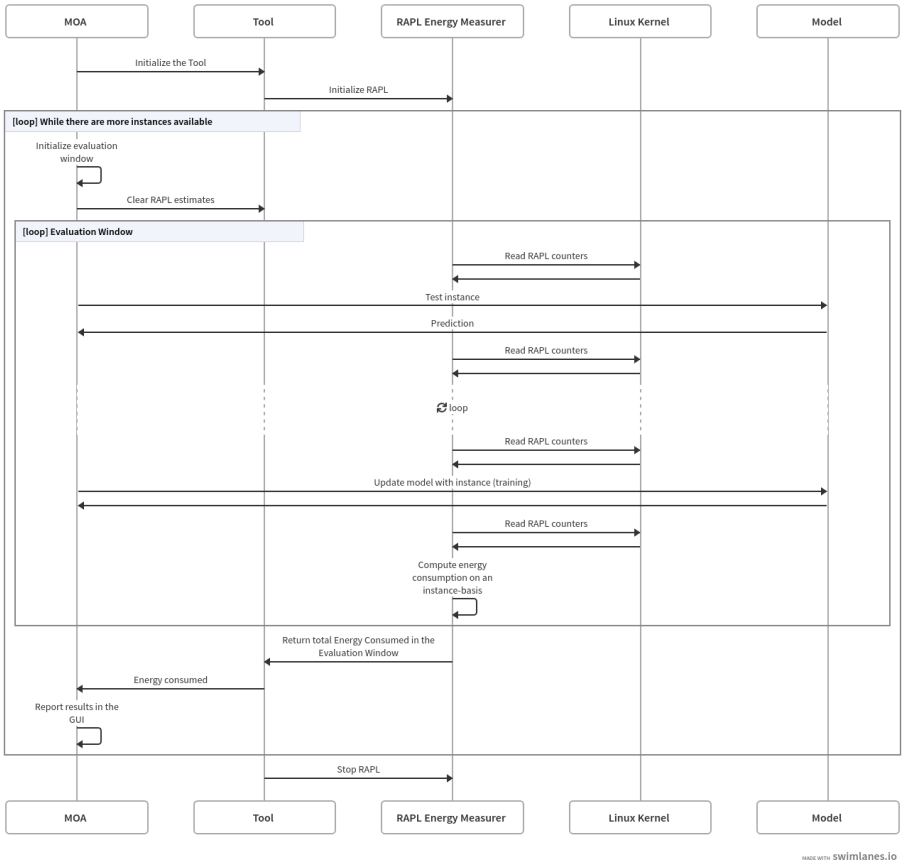

Fig. 2. Measuring energy consumption using RAPL.

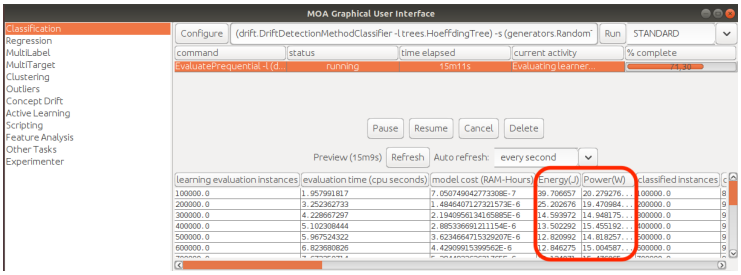

Fig. 3. Screenshot of the Massive Online Analysis (MOA) framework with results provided by the proposed tool.

5 Experiments

This section presents the experiments conducted to assess our proposed tool to measure energy consumption in data streams. In particular, this section is divided into two experiments. First, we analyze our tool against a hardware solution, which serves as a gold standard for the energy consumed. Next, we analyze the energy consumed by different classifiers in Prequential test-then-train validation in stationary and non-stationary scenarios. All tests were performed on a desktop computer running Ubuntu Desktop 18.04 LTS; Intel Core i7-2600 Sandy Bridge CPU; 4GB RAM, and 250 GB HDD Hard Drive.

5.1 Experiment 1 - Validation against a hardware solution

The first experiment conducted aimed to assess whether the energy consumption measurement via hardware and software solutions were equivalent or, at least, correlated. In this experiment, both software and hardware TP-LINK HS 110 wall plug solutions were connected to the computer, and a CPU stressor called stress-ng [23] was used to quantify energy consumption rates.

Initially, it was deemed necessary to assess whether the measurement of energy consumption by hardware or software is equivalent and when using only one software tool (the plugin) would not compromise the results of this study. The stressor was configured to generate a 10% stress level for 10 minutes and progressively perform increments by 10% until 100% stress was reached. Once 100% stress was reached, the test was incremented to use an extra processor core. This process was repeated until all four cores were allocated. The entire testbed encompassed five runs, and the average results are given in Figure 4. Both lines indicate the various stress levels on the computer, with the blue line relating to the software power consumption measurement and the green line the hardware results.

These results depict that the software solution does not meet the hardware solution readings. These results are expected since the hardware readings account for the entire computer, i.e., the operational system and other software that is being run; thus, it is reasonable to assume that the stressor occupies a part of the overall energy being consumed. Even though it is clear that the results do not match, they possess a 99.97% correlation, which depicts a strong correlation. Therefore, we verify that although the software solution does not accurately describe a computer's total energy consumption, its results correlate with the actual consumption measured from a hardware tool, as also observed in [8, 27].

5.2 Experiment 2 - Analyzing Different Classifiers in Stationary and Non-Stationary Environments

In this experiment, we used our proposed tool to quantify the energy consumption of different classifiers in stationary and non-stationary environments. In particular, synthetic data streams were created using the Massive Online Analysis (MOA) framework using the Agrawal (AGR) [1], Assets Negotiation (AN)

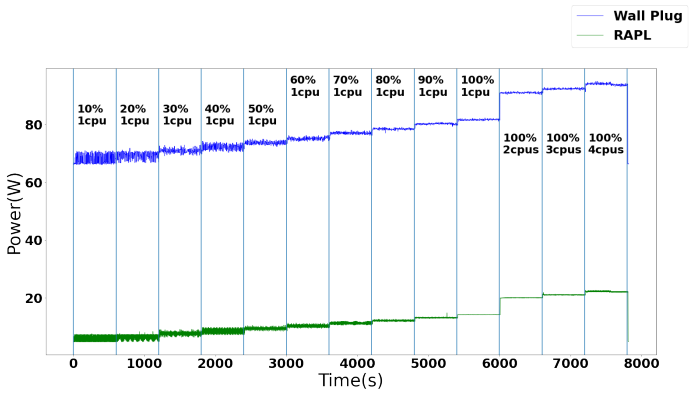

Fig. 4. Comparison of the energy consumption quantified by hardware and software.

[2], and SEA [29] generators. Each stream possessed 1 million instances, and two variants were created, one with concept drifts located at the middle of the experiment (drift position equal to 500,000) and another without concept drift. Experiment variants marked with a **-D** suffix stand for drifting experiments. The data streams mentioned above were used to assess Naive Bayes (NB), Hoeffding Tree (HT), Hoeffding Adaptive Tree (HAT), and Adaptive Random Forest (ARF) classifiers. All classifiers used the default parameters available in MOA, except for ARF, which used 100 ensemble members (each with an individual thread) and a grace period $n_{min} = 50$. All experiments were conducted using the Prequential validation scheme proposed in [11], i.e., each instance was retrieved and used for testing and training. The source code to reproduce this experimentation is also available in the code repository.

Discussion The results obtained are given in Tables 1, 2, 3, and 4. These tables provides accuracy, processing time (in seconds), memory consumption (in GB-Hours), and energy consumption rates (in Watts), respectively. First, we highlight that Naive Bayes (NB) has the lowest accuracy rates in all scenarios. This result corroborates that decision trees and their ensembles are more interesting when this particular evaluation metric is pursuit. We highlight, for instance, the results obtained by ARF in drifting experiments, which are expected since it has multiple learners coupled with drift detectors to detect and adapt to such changes. Nonetheless, decision trees bring forward computational overheads that are quantified by the remainder of the metrics. First, we see that Hoeffding Tree (HT), Hoeffding Adaptive Tree (HAT), and the Adaptive Random Forest (ARF) are slower than Naive Bayes (NB). The processing times for HT, HAT, and ARF are 7, 11, and 44,751 times slower than NB, respectively. Similar results are observed for RAM consumption, in which HT, HAT, and ARF consume more RAM than NB. Again, we highlight the RAM consumption observed by ARF, which is approximately 10^8 times higher than its counterparts. Finally,

Table 1. Accuracy results obtained during experimentation.

Experiment	Accuracy (%)			
	NB	HT	HAT	ARF
AGR	94.39	**94.97**	94.51	94.52
AGR-D	75.67	85.72	87.63	**90.66**
AN	92.36	**94.87**	94.83	94.86
AN-D	83.95	94.53	94.71	**94.73**
SEA	86.95	89.39	89.40	**89.67**
SEA-D	87.02	88.83	89.14	**89.55**

Table 2. Memory consumption results obtained during experimentation.

Experiment	Memory consumption (GB-Hours)			
	NB	HT	HAT	ARF
AGR	$\mathbf{1.60\times10^{-9}}$	6.64×10^{-6}	7.87×10^{-6}	7.72×10^{1}
AGR-D	$\mathbf{1.78\times10^{-9}}$	1.25×10^{-5}	7.85×10^{-6}	12.40×10^{1}
AN	$\mathbf{1.21\times10^{-8}}$	4.72×10^{-6}	2.02×10^{-5}	4.38×10^{1}
AN-D	$\mathbf{9.51\times10^{-9}}$	6.81×10^{-6}	8.13×10^{-6}	3.78×10^{1}
SEA	$\mathbf{3.59\times10^{-10}}$	6.11×10^{-7}	6.45×10^{-6}	5.99×10^{1}
SEA-D	$\mathbf{4.31\times10^{-10}}$	6.33×10^{-7}	2.27×10^{-6}	9.12×10^{1}

the energy consumption values depict that NB is the less-consuming algorithm, which the exception of HAT in the AGR-D experiment. These results are expected since NB is much faster and less memory-consuming; thus, less energy is required to finalize an experiment. Regarding the AGR-D experiment, it is relevant to emphasize that HAT's energy consumption has decreased since it has a drift detector that restarted the entire tree learning process, i.e., it replaced the entire tree with a single decision stump, and thus, its computational cost after the drift has greatly decreased. This is a relevant scenario in which energy consumption is not directly related to processing time and memory consumption, and it allows a better analysis by researchers and practitioners on which classifier should be used in a specific scenario. Focusing on ARF, we also highlight that the energy consumption rates are not directly related to either processing time and memory consumption rates since, despite taking much more time and memory to run, its energy consumption was roughly twice when compared to its counterparts. This result can be explained due to ARF's implementation, which is multi-threaded, and even by combining 100 learners the energy consumption is not 100 times greater than its counterparts.

6 Conclusion

In this work, we brought forward a tool for quantifying energy consumption in data stream mining. Our tool was embedded within the Massive Online Analysis (MOA) framework, thus allowing researchers to rapidly quantify the energy consumption of different classification methods under different streaming settings

Table 3. Processing time results obtained during experimentation.

Experiment	Processing time (s)			
	NB	HT	HAT	ARF
AGR	**1.35**	16.57	19.24	74639.15
AGR-D	**1.50**	21.42	18.99	49600.75
AN	**16.28**	24.21	35.69	22538.36
AN-D	**12.80**	24.14	25.19	20149.68
SEA	**0.75**	5.72	16.94	96930.68
SEA-D	**0.90**	5.92	11.81	43513.95

Table 4. Energy consumption results obtained during experimentation.

Experiment	Energy (W)			
	NB	HT	HAT	ARF
AGR	**34334.04**	37452.53	35283.41	66811.31
AGR-D	37747.63	37435.13	**36280.75**	71646.43
AN	**32520.12**	33972.08	34425.10	84459.67
AN-D	**33414.73**	34277.01	34851.68	67507.25
SEA	**25749.19**	38479.07	35183.48	69747.71
SEA-D	**33910.95**	38363.17	35245.82	55118.25

and validation processes. To validate our proposal, we first conducted a testbed using a processor stressor to compare the proposed software readings against a hardware wall plug. Next, we tested different classifiers in stationary and drifting scenarios in a Prequential validation scheme. Results showed that our tool allows the identification of energy consumption rates of different classifiers under different scenarios, i.e., drifting and non-drifting data streams. The energy consumption rates allow a more fine-grained analysis of the classifiers as energy consumption is not directly tied with processing time and memory consumption, especially under concept drifting scenarios and multi-threading implementations.

In future works, we plan to extend our tool to encompass different classification, regression, and clustering validation schemes, including more datasets and scenarios. We also plan to port our tool to Python-based frameworks, such as River [25]. Finally, we also plan to make our tool available in AMD and Apple's ARM platforms and add support to GPU consumption since neural networks are increasingly used in streaming settings [20].

References

1. Agrawal, R., Imielinski, T., Swami, A.: Database mining: A performance perspective. IEEE transactions on knowledge and data engineering **5**(6), 914–925 (1993)
2. Barddal, J.P., Murilo Gomes, H., Enembreck, F., Pfahringer, B., Bifet, A.: On dynamic feature weighting for feature drifting data streams. In: Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part II 16. pp. 129–144. Springer (2016)

3. Bifet, A., Gavalda, R.: Learning from time-changing data with adaptive windowing. In: Proceedings of the 2007 SIAM international conference on data mining. pp. 443–448. SIAM (2007)
4. Bifet, A., Gavalda, R.: Adaptive learning from evolving data streams. In: Advances in Intelligent Data Analysis VIII: 8th International Symposium on Intelligent Data Analysis, IDA 2009, Lyon, France, August 31-September 2, 2009. Proceedings 8. pp. 249–260. Springer (2009)
5. Bifet, A., Gavalda, R., Holmes, G., Pfahringer, B.: Machine learning for data streams: with practical examples in MOA. MIT press (2023)
6. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: Moa: Massive online analysis. Journal of Machine Learning Research **11**(52), 1601–1604 (2010), http://jmlr.org/papers/v11/bifet10a.html
7. David, H., Gorbatov, E., Hanebutte, U.R., Khanna, R., Le, C.: Rapl: Memory power estimation and capping. In: Proceedings of the 16th ACM/IEEE International Symposium on Low Power Electronics and Design. p. 189–194. ISLPED '10, Association for Computing Machinery, New York, NY, USA (2010). https://doi.org/10.1145/1840845.1840883, https://doi.org/10.1145/1840845.1840883
8. Desrochers, S., Paradis, C., Weaver, V.M.: A validation of dram rapl power measurements. In: Proceedings of the Second International Symposium on Memory Systems. p. 455–470. MEMSYS '16, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2989081.2989088, https://doi.org/10.1145/2989081.2989088
9. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. p. 71–80. KDD '00, Association for Computing Machinery, New York, NY, USA (2000). https://doi.org/10.1145/347090.347107, https://doi.org/10.1145/347090.347107
10. Gama, J.: Knowledge Discovery from Data Streams. Chapman & Hall/CRC, 1st edn. (2010)
11. Gama, J., Sebastião, R., Rodrigues, P.P.: On evaluating stream learning algorithms. Machine Learning **90**(3), 317–346 (Mar 2013). https://doi.org/10.1007/s10994-012-5320-9
12. Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation (2014). https://doi.org/10.1145/2523813
13. Garcia-Martin, E., Lavesson, N., Grahn, H.: Energy efficiency analysis of the very fast decision tree algorithm. Trends in Social Network Analysis: Information Propagation, User Behavior Modeling, Forecasting, and Vulnerability Assessment pp. 229–252 (2017)
14. Garcia-Martin, E., Lavesson, N., Grahn, H.: Identification of energy hotspots: A case study of the very fast decision tree. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 10232 LNCS, pp. 267–281. Springer Verlag (2017). https://doi.org/10.1007/978-3-319-57186-7_21
15. García-Martín, E., Lavesson, N., Grahn, H., Casalicchio, E., Boeva, V.: How to Measure Energy Consumption in Machine Learning Algorithms. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 11329 LNAI, pp. 243–255. Springer Verlag (sep 2019). https://doi.org/10.1007/978-3-030-13453-2_20

16. García-Martín, E., Rodrigues, C.F., Riley, G., Grahn, H.: Estimation of energy consumption in machine learning. Journal of Parallel and Distributed Computing **134**, 75–88 (Dec 2019). https://doi.org/10.1016/j.jpdc.2019.07.007
17. García-Martín, E., Bifet, A., Lavesson, N.: Energy modeling of hoeffding tree ensembles. Intelligent Data Analysis (2020)
18. García-Martín, E., Bifet, A., Lavesson, N.: Green accelerated hoeffding tree (2020), http://urn.kb.se/resolve?urn=urn:nbn:se:bth-19152
19. Gomes, H.M., Bifet, A., Read, J., Barddal, J.P., Enembreck, F., Pfharinger, B., Holmes, G., Abdessalem, T.: Adaptive random forests for evolving data stream classification. Machine Learning **106**(9), 1469–1495 (Oct 2017). https://doi.org/10.1007/s10994-017-5642-8, https://doi.org/10.1007/s10994-017-5642-8
20. Gunasekara, N., Gomes, H.M., Pfahringer, B., Bifet, A.: Online hyperparameter optimization for streaming neural networks. In: 2022 International Joint Conference on Neural Networks (IJCNN). pp. 1–9. IEEE (2022)
21. Hähnel, M., Döbel, B., Völp, M., Härtig, H.: Measuring energy consumption for short code paths using rapl. ACM SIGMETRICS Performance Evaluation Review **40**(3), 13–17 (2012)
22. Hoeffding, W.: Probability inequalities for sums of bounded random variables. The collected works of Wassily Hoeffding pp. 409–426 (1994)
23. King, C.I.: Stress-ng. URL: http://kernel. ubuntu. com/git/cking/stressng. git/(visited on 28/03/2018) p. 39 (2017)
24. Klawonn, F., Angelov, P.: Evolving extended naive bayes classifiers. In: Sixth IEEE International Conference on Data Mining-Workshops (ICDMW'06). pp. 643–647. IEEE (2006)
25. Montiel, J., Halford, M., Mastelini, S.M., Bolmier, G., Sourty, R., Vaysse, R., Zouitine, A., Gomes, H.M., Read, J., Abdessalem, T., et al.: River: machine learning for streaming data in python. The Journal of Machine Learning Research **22**(1), 4945–4952 (2021)
26. Noureddine, A.: Powerjoular and joularjx: multi-platform software power monitoring tools. In: 2022 18th International Conference on Intelligent Environments (IE). pp. 1–4. IEEE (2022)
27. Phung, J., Lee, Y.C., Zomaya, A.Y.: Modeling system-level power consumption profiles using RAPL. In: NCA 2018 - 2018 IEEE 17th International Symposium on Network Computing and Applications. Institute of Electrical and Electronics Engineers Inc. (nov 2018). https://doi.org/10.1109/NCA.2018.8548281
28. Shao, Y.S., Brooks, D.: Energy characterization and instruction-level energy model of intel's xeon phi processor. In: International Symposium on Low Power Electronics and Design (ISLPED). pp. 389–394. IEEE (2013)
29. Street, W.N., Kim, Y.: A streaming ensemble algorithm (sea) for large-scale classification. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 377–382 (2001)
30. Terpstra, D., Jagode, H., You, H., Dongarra, J.: Collecting performance data with papi-c. In: Tools for High Performance Computing 2009: Proceedings of the 3rd International Workshop on Parallel Tools for High Performance Computing, September 2009, ZIH, Dresden. pp. 157–173. Springer (2010)
31. Widmer, G., Kubat, M.: Learning in the presence of concept drift and hidden contexts. Machine learning **23**, 69–101 (1996)