Energy Efficiency Analysis of the Very Fast Decision Tree Algorithm

Eva Garcia-Martin, Niklas Lavesson, and Håkan Grahn

Abstract Data mining algorithms are usually designed to optimize a trade-off between predictive accuracy and computational efficiency. This paper introduces energy consumption and energy efficiency as important factors to consider during data mining algorithm analysis and evaluation. We conducted an experiment to illustrate how energy consumption and accuracy are affected when varying the parameters of the Very Fast Decision Tree (VFDT) algorithm. These results are compared with a theoretical analysis on the algorithm, indicating that energy consumption is affected by the parameters design and that it can be reduced significantly while maintaining accuracy.

Key words: Energy efficiency, Green computing, Very Fast Decision Tree, Big Data

1 Introduction

Data stream mining is gaining importance with the evolution of hardware, sensor systems and technology. The rate at which data is generated is increasing day by day, challenging storage and computational efficiency [30]. Digital Universe Study [2] has predicted that by 2020, 40,000 exabytes of data will be processed, most of them

Eva Garcia-Martin
Blekinge Institute of Technology, SE-371 79, Karlskrona, Sweden, e-mail: eva.garcia.martin@bth.se

Niklas Lavesson
Blekinge Institute of Technology, SE-371 79, Karlskrona, Sweden, e-mail: niklas.lavesson@bth.se

Håkan Grahn
Blekinge Institute of Technology, SE-371 79, Karlskrona, Sweden, e-mail: hakan.grahn@bth.se

originating from devices that automatically generate data. Many algorithms in data stream mining are designed to process fast and potentially infinite streams [25, 38].

Traditionally, the machine learning community has considered accuracy as the main factor when building algorithms. With the appearance of big data analytics and data stream mining, scalability has also been a key factor to consider. In this context, scalability stands for how fast an algorithm can process the incoming data. The problem that we address in this study is the fact that few researchers in the data mining community consider energy consumption as an important measure.

It has been shown that energy consumption can be reduced in every layer of the Open Systems Interconnection (OSI) model [51, 56]. Hardware solutions to reduce energy consumption have been focused on, e.g. using the Dynamic Voltage Frequency Scaling (DVFS) technique and on parallel computing [48, 44]. During recent years, the interest in developing energy efficient software solutions has increased significantly, leading to a creation of applications to measure energy consumption in software [47].

This paper introduces energy consumption and energy efficiency as important factors to consider during data mining algorithm analysis and evaluation, and to demonstrate the use of these factors in a data stream mining context. The consideration of energy efficiency can help companies and researchers move towards green computing [37] while improving the business profits.

Social networks is a very good example of a domain in need of efficient data processing and analysis of algorithms. Companies such as Facebook, Twitter, and Instagram, rely on large data clusters consuming vast amounts of energy. Facebook, for example, generates 4 million posts every minute [9], which creates interesting challenges for algorithms that are running continuously in their network. One of these challenges is to optimize such algorithms in terms of energy consumption. Even a small improvement will reduce energy on a large scale, due to the nature of the network.

We conducted an experiment to illustrate a possible scenario where energy consumption is relevant to study. More specifically, we studied how energy and accuracy are affected by changing the parameters of the VFDT (Very Fast Decision Tree) algorithm [25]. We make a comparison between the theoretical analysis on the algorithm and the experimental results, which indicate that it is possible to significantly reduce the energy consumption of the VFDT algorithm while maintaining similar levels of accuracy. The main contribution of this paper is the introduction of energy consumption as a key factor to consider in data mining algorithms. This is supported by a theoretical and empirical analysis that illustrate an example on how to build sustainable and efficient algorithms and the reasons behind energy consumption.

This paper is an extension of the paper titled: *Energy Efficiency in Data Stream Mining* [45]. While such publication centers on observing the energy consumption only from an empirical perspective, this study motivates the experimental setup and results by analyzing the behavior of the VFDT from a theoretical and empirical perspective. Therefore, we can compare how the algorithm behaves in reality from what we predicted theoretically. On top of that, more relevant parameters have been chosen and two real world datasets have been added to the experiment.

2 Background

In this section we first explain the importance of energy consumption in data mining. Then, we briefly explain data stream mining and why it is different from standard data mining, and finally we introduce some terminology related to power, energy, energy efficiency and computational efficiency.

2.1 Energy-awareness

The demand for energy is increasing day by day [51]. World leaders and scientists focus on finding a solution towards this problem, centering on two key factors: developing new sources of clean energy and decreasing energy usage [22] [13], which would lead to a reduction in CO_2 emissions. The main reason why researchers and every citizen should be aware of energy consumption is because energy pollutes. Every device that we use in a daily bases that consumes energy produces CO_2. Nowadays, based on a study conducted by the World Health Organization, air pollution kills more people than malaria and aids combined [46]. This argument is based on what is known as ecological or environmental footprint [3], that measures how much impact a certain person or action has in relation to the environment. For instance, carbon footprint measures how many greenhouse gases are produced by an individual or event, expressed as CO_2 [1]. Therefore, if companies and individuals are aware of the footprint of their computations, their impact could be reduced by making them energy efficient.

There have been studies that measure the environmental impact of queries in search engines [11]. Considering that there are approximately 66k Google queries per second [7], reducing the CO_2 emissions of search queries will significantly impact the environment. If we translate this example to data stream mining, we can picture the execution of data stream mining algorithms in servers running during 24 hours a day, for a complete year. In this case, building energy-aware algorithms has the following consequences:

- Reduction of CO_2 emissions to the atmosphere.
- Reduction of air pollution, therefore reducing the number of deaths per year due to this matter.
- Reduction of the money spent on energy.
- Increase of the battery life of mobile devices and sensor networks, if the algorithm is implemented in such contexts.

2.2 Data Stream Mining

Data stream mining is the process of building models by exploring and extracting patterns from a stream of data.

The core assumption of data stream mining, in comparison to data mining, is that the examples are inspected only once, so we have to assume that if they are not processed immediately they are lost forever [50, 20]. Moreover, it is considered that the data arrives online, with no predefined order, at a high-speed and with time-changing characteristics. Data stream mining algorithms should be able to process potentially infinite streams while updating the model incrementally [31, 38].

2.3 Terminology

In this section we clarify several concepts related to energy, power, and efficiency. Energy is a measurement of the amount of *fuel* used for a specific application. It is measured in Joules (J) or kWh. Power is a measurement of the rate at which energy is consumed. It is measured in Joules/second, which is equal to Watts (W). The following is an example that illustrates the relationship between power and energy: A process is running for 3.94 seconds consuming an estimate power of 1.81 W. The total energy consumed is: $3.94 \times 1.8 = 7.092J = Ws = 1.99 \times 10^{-3}$ Wh.

Energy efficiency has a specific definition at Green500 [8], being, *The amount of operations per watt a computer can perform.* This definition is related to hardware. In this study, whenever we mention energy efficiency we refer to reducing the energy consumption of some process or algorithm.

In theoretical machine learning, researchers introduced the computational learning theory [40], where they analyze the computational complexity of algorithms. They approach computational efficiency as a way of designing less computationally complex algorithms that can run in polynomial time.

3 Related Work

In this section we first review literature related to energy awareness in software and hardware. Then, we examine relevant work in the data stream mining field, focusing on the VFDT algorithm. Finally, we review papers that are related to both energy consumption and data stream mining.

Research in energy awareness at the software level started many years ago, when researchers began to realize the importance of the energy consumed by a software application. In 1994, the first systematic attempt to model the power consumption of the software components of a system was presented [53]. After that, in 1999, PowerScope was presented [27], a software tool for profiling the energy usage of applications. The novelty of this approach is that energy consumption can be mapped to program structure to analyze which procedures consume more energy. Companies such as Microsoft [5] and Intel [6] have invested in developing software tools to help developers reduce the energy consumption of their applications. During the past years, the Spiral research group [12] has gained interest in building energy efficient software. They show that energy consumption depends not only on the time and number of computations, but also on the stress of the processor, the I/O operations and many other factors. They have developed a software tool, PowerAPI, where they show that they can get high accurate modeling of the power consumption of software applications. They have evaluated their model by comparing their results and method with hardware power meters obtaining promising results [21, 47].

In relation to energy efficiency at the hardware level, one of the most important techniques, implemented in most contemporary processors, is Dynamic Voltage Frequency Scaling (DVFS). DVFS is a power saving technique used and improved by many researchers. One improvement is Real Time DVFS, an implementation of DVFS for real time systems [48]. Another area that is gaining importance nowadays is parallel computing, where there are relevant energy savings by employing more cores on a processor [44]. Several energy-saving approaches, such as *Cost optimization for power-aware computing* have been developed in the past years [51].

In relation to data stream mining, researchers have developed efficient approaches to mine data streams, as outlined below. There have been several reviews conducted in data stream mining since 2005. Two general reviews [30, 15], portray techniques and concepts such as data-based techniques, task-based techniques, data stream classification and frequent pattern mining. More specific reviews center on topics such as sensor networks [28] and knowledge discovery [31].

From the reviews explained above, we have extracted six main techniques and approaches in data stream mining: Data stream clustering [35], Data stream classification [25], Frequent Pattern Mining [17], Change Detection in data streams [14, 41], Sliding window techniques [23] and Stream mining in sensor networks [34, 28]. We have decided to focus in Data Stream classification and change detection in data streams.

Concept drift refers to a change between the input data and the target variable on an online supervised learning scenario. The first framework that dealt with concept drift was proposed to also address efficiency and robustness [55]. Nowadays, researchers consider concept-drift as an important aspect when building algorithms for other specific purposes. A survey on different methods that address concept drift has been conducted in 2014 [33].

Classification is considered a challenging problem in a data stream mining scenario [15]. The main reason is that many of the traditional classification techniques and algorithms were designed to build models from static data.

One of the key breakthroughs in supervised online learning was made with the development of the Hoeffding Tree algorithm and the Very Fast Decision Tree (VDFT) learner [25]. In contrast to previous algorithms, such as as SPRINT [52] and ID5R [54], this new approach was able to deal with potential infinite streams, arriving at a fast pace and with low computational cost. The VFDT learner is able to process examples at a high rate in constant time. One year later, the same authors created a new version of the VDFT algorithm, CVFDT, that was able to adapt to concept-drift [38]. Another extension on the VFDT algorithm appeared two years later, with a new decision tree learner that could efficiently process numerical attributes [39]. In the same line, a decision tree algorithm was created for spatial data streams [24]. We would like to mention relevant methods that address different classification problems, namely: On-Demand classification [16, 14], Online Information Network (OLIN) [42], LWClass [29], ANNCAD [43] and SCALLOP [26].

In relation to energy awareness in data stream mining, several researchers have conducted studies where they emphasize the importance of energy consumption [32, 30, 18]. While the first two are concerned on energy savings for sensor networks, the second one centers on examine the energy consumption of different data analysis techniques. To the best of our knowledge, the last work is the one most related to ours.

We can observe that there is no specific research on making energy consumption a key factor on data stream mining, since the research has been centered towards specific applications or hardware modifications. We would like to change this approach by proposing energy consumption as the new factor to consider when building, optimizing or creating new algorithms in data stream mining. We believe that this is the next natural step to take, since other researchers in similar fields, hardware and software, have already taken that step.

4 Theoretical analysis

This section aims to theoretically analyze the behavior of the Very Fast Decision Tree (VFDT) algorithm [25]. VFDT is an online decision tree algorithm able to build a decision tree from a stream of data by analyzing the data sequentially and only once.

The decision tree is built sequentially, where the tree waits until it gathers enough examples or instances from the stream. After those n instances arrive, the algorithm analyzes them and obtains the best attribute to split the tree on. The key feature is to obtain the optimal value of n that will split in the same attribute as if we had all examples available to analyze. To obtain the first best value of n, the authors make use of the Hoeffding Bound [36], represented by ε in Equation 1.

$$\varepsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}} \tag{1}$$

This bound states that with probability 1-δ, the chosen attribute at a specific node after seeing n number of examples, will be the same attribute as if the algorithm had seen infinite number of examples. Therefore, δ represents one minus the probability of choosing the correct attribute to split on. The reason is that there will be no split on a certain attribute unless $\Delta\overline{G} > \varepsilon$. $\Delta\overline{G}$ stands for the difference in information gain between the best two attributes. Thus, if the number of examples n is small,

ε will be high, making it harder to split on an attribute unless $\Delta\overline{G}$ is big enough, meaning that there is a clear attribute that is the winner. Based on the equation, whenever we see more examples, n increases, making ε smaller and then making it easier to split on the top attribute. The reasoning behind this is that whenever we see more examples we are more confident on the split. In order to speed up the computations, some parameters are introduced, that will make the algorithm behave in a slightly different way that the one currently explained.

The next paragraphs theoretically analyze how accuracy and energy would differ when varying the different parameters of the VFDT algorithm. The chosen parameters to be varied are: *nmin*, τ, δ, *memory limits*, *memory management*, *split criterion* and *poor attributes removal*. As a general observation, the theoretical analysis made about the parameters can not be generalized to all cases, since it would vary depending on the input data. For that reason, the assumptions that are made in the following paragraphs, are based on the reasoning and experiments from the original paper by the authors.

nmin parameter is the minimum number of examples that the algorithm must see before calculating ε to check if there are sufficient statistics for a good split. The authors introduce this parameter to reduce execution time and computational effort when building the tree, since it is very unlikely that after just one instance the algorithm has a more convincing split. The default value of *nmin* is 200. If the value of *nmin* increases, then accuracy will be slightly reduced, since the tree will have a lower number of nodes. From the original paper, the difference in accuracy was of a merely 1.1%, but the execution time was 3.8 times faster when using and increasing *nmin*. Therefore, we predict that the energy would decrease when increasing *nmin*, since we would be decreasing the computational effort and time to analyze each instance.

τ parameter represents the tie breaking parameter. Whenever the difference between both attributes is small enough, that means both attributes are equally good, making no sense to wait a longer time for more examples to make a split. The absence of this parameter has been shown to decrease accuracy, since the decision tree contains fewer nodes in it. However, being able to make more splits on the data allows to obtain a finer grained decision tree. In an extreme situation where both attributes are exactly the same, the tree would stall, failing to grow. So increasing τ could, in an ideal scenario with an infinite stream of data, increase accuracy and decrease energy, since the tree is built before, reducing the time and the number of computations [20]. But if we analyze the same amount of examples, then increasing τ could increase the energy, due to the fact that with lower τ we make less number of computations.

δ parameter represents one minus the probability of choosing the correct attribute to split on. If δ increases then the desired probability is smaller. Hence, the tree will grow faster, having more nodes. Since the difference on nodes between a higher and a lower δ will not be as high as when removing τ, and the probability of a correct split is lower, our assumption is that the accuracy will be lower when δ increases. At the same time, if δ decreases, then the probability of making a correct split increases, increasing accuracy. In terms of energy, we believe that the power and time consumed to build the tree will vary depending on the incoming data. We hypothesize that with a lower δ there will be more power spent on computing information gain rather than in building nodes. However we currently do not have the knowledge of which consumes more power. At the same time, the tree could be built faster since we are spending less time on building the tree nodes.

In terms of *memory limits* and *memory management*, we predict that with a lower memory limit, the tree will induce fewer nodes, having less energy consumption and less accuracy. At the same time, the tree uses pruning techniques to reduce the memory spent on building the tree, removing less promising leaves, which in some cases could improve accuracy. It will depend on how different is the memory limit and how many nodes differ from each setting to correctly predict if the accuracy will be lower or higher. In a realistic case we assume that limiting the memory consumption and the tree growth will decrease accuracy and energy.

The default split criterion used by the authors in this algorithm is *information gain*. We have tested also *Gini index*. From a theoretical aspect, it has been shown [49] that it is not conclusive which of the two criterion will perform a better job in general, since they both have shown similar results, differing only by 2%.

The last parameter that is going to be modified is *removing poor attributes*. This parameter aims to analyze attribute performance to find attributes that perform poorly and which are very unlikely to be chosen for a further split. The process that the authors follow is by analyzing the information gain of all attributes in every split, and when this value, for a specific attribute, is less than the information gain of the best attribute by more than a difference of ε, then the attribute is ignored for that leaf. In theory, this method should increase accuracy and decrease the amount of computations.

Table 1 represents a summary of all the predictions of the parameters variations explained above. It shows how energy, accuracy and tree size will vary when increasing or decreasing the mentioned parameters.

5 Experimental Design

5.1 Problem Definition

In order to empirically study the different parameters of the VFDT algorithm we have created an experiment where we vary the parameters theoretically analyzed in Section 4. This experiment aims to evaluate the performance of the algorithm under different setups in terms of accuracy, energy, execution time and size of the tree. An implicit goal is understanding why varying the parameters in a certain way increases or decreases accuracy and energy, and if it matches with the theoretical reasoning. The experiment has three phases. First, we obtain the datasets, then we input them into the algorithm under different setups, and we finally evaluate the performance

Table 1 Summary of the theoretical behavior of the parameters

PARAMETER	MODIFICATION	ACC	ENERGY	NODES
nmin	Increase	Lower	Lower	Fewer
	Decrease	Higher	Higher	More
τ	Increase	Higher	Higher	More
	Decrease	Lower	Lower	Fewer
δ	Increase	Lower	†	More
	Decrease	Higher	†	Fewer
MEM1	100KB	Lower	Lower	Fewer
	2GB	Higher	Higher	More
MEM2	ON	Lower	Lower	Fewer
SPLT CRIT	S2	†	†	†
RPA	ON	Higher	Lower	Fewer

†=The variation depends on the input data.

of each model in terms of accuracy and energy consumption. The way to measure energy is explained in Section 5.3.

5.2 Data Gathering

We have gathered four different datasets to perform this experiment. Two datasets are synthetically generated and the other two are real world datasets. The main difference between synthetic and real world datasets, is that the first ones are randomly generated based on a specific function and distribution, and the second ones are representations of some measure that exists in reality. The idea is to show that the solution proposed in this paper of analyzing energy consumption of algorithm, applies to both real world and controlled environment. It improves the generalizability of the results.

The synthetic datasets have been generated with MOA (Massive Online Analysis) [19], and the functions: *Random Tree Generator*, *Hyper Plane generator*. The random tree function generates a tree as explained by the authors of the VFDT algorithm [25]. We have chosen this dataset because is the same dataset that the authors of the VFDT use in their experiments, so we consider it as a baseline of a standard behavior of the algorithm. Then we chose a more challenging synthetic dataset, since the hyperplane generator is often used to test algorithms that can handle concept drift, such as CVFDT. Even though this algorithm is not developed to handle concept drift, we wanted to test the different setups in a completely different dataset than the random tree. The Hyper plane generator uses a function to generate data in the form of a plane in d dimensions [38]. The orientation of the hyperplane can easily be varied by adjusting its weights, creating different concepts. Depending on the coordinates of the plane, the examples are labeled as negative or positive. All synthetic generators have generated a total of 1 million instances, and we have chosen the number of numerical and nominal attributes based on the default settings in MOA. The tree-generated dataset is a binary classification dataset with 5 nominal and 5 numerical attributes. The hyperplane-generated dataset is also a binary classification dataset with 10 different attributes and 2 concept drift attributes.

Since usually synthetic datasets do not have the same properties as real world datasets, we have decided to add two real world datasets to the experiment. The first one represents instances that try to predict good poker hands based on a given hand [10]. There are a total of 1,025,010 instances and 11 attributes and has been normalized by the MOA researchers. The second real world dataset is the normalized airline dataset, created by Elena Ikonomovska [4]. This classification dataset classifies flights into delayed or not depending on the route of the flight, based on the departure and arrival airports. It contains a total of 8 attributes, being: Airline, flight number, origin, destination, day of the week, elapsed time, duration of the flight, and if there was a delay or not. The motivation behind these datasets is the amount of instances available, making them perfect candidates to test data stream mining algorithms. Table 2 summarizes the information from the different datasets, regarding the number of instances, attributes and type of the dataset.

Table 2 Datasets summary

Dataset	Name	Type	Instances	Nominal attributes	Numeric attributes
1	Random tree	Synthetic	1,000,000	5	5
2	Hyperplane	Synthetic	1,000,000	1	9
3	Poker	Real world	1,025,010	6	5
4	Airlines	Real world	539,383	5	3

5.3 Methodology

This section aims to explain the settings of the experiment, the parameters varied and the tools used to perform it.

5.3.1 Parameter choice

We have chosen to vary the parameters explained in Section 4, namely: *nmin*, τ, δ, *memory limits*, *memory management*, *split criterion*, and *removing poor attributes*. *nmin* was varied from the default value, 200, to a maximum value of $1,700$ with steps of 500. τ was varied from the default value, 0.05, to a maximum value of 0.13, a minimum value of 0.01 and with steps of 0.04. δ was varied from the default value, 10^{-7}, to a maximum and minimum values of 10^{-1} and 10^{-10}, respectively. The step is of 10^{-3}. Memory limit varied from 100KB, to 30MB (default value) until 2GB, that was the maximum allowed by MOA (Massive Online Analysis), the tool that will be furthered explained. The memory management and removing poor attributes were tested by activating and deactivating them. Finally, Gini index was tested against Information Gain. Every parameter was varied while maintaining the other parameters constant, in their default value. The aim is to understand the behavior of each parameter on its own, without having external interference with the rest of the parameters. A summary of the parameters setup is shown in Table 3.

Table 3 Parameter Configuration Index. Different configurations of the VFDT algorithm.

IDX	*nmin*	τ	δ	MEM1	MEM2	S.CRT	RPA
A	200	0.05	10^{-7}	30MB	No	S1	No
B	**700**	0.05	10^{-7}	30MB	No	S1	No
C	**1,200**	0.05	10^{-7}	30MB	No	S1	No
D	**1,700**	0.05	10^{-7}	30MB	No	S1	No
E	200	**0.01**	10^{-7}	30MB	No	S1	No
F	200	**0.09**	10^{-7}	30MB	No	S1	No
G	200	**0.13**	10^{-7}	30MB	No	S1	No
H	200	0.05	$\mathbf{10^{-1}}$	30MB	No	S1	No
I	200	0.05	$\mathbf{10^{-4}}$	30MB	No	S1	No
J	200	0.05	$\mathbf{10^{-10}}$	30MB	No	S1	No
K	200	0.05	10^{-7}	**100KB**	No	S1	No
L	200	0.05	10^{-7}	**2GB**	No	S1	No
M	200	0.05	10^{-7}	30MB	**Yes**	S1	No
N	200	0.05	10^{-7}	30MB	No	**S2**	No
O	200	0.05	10^{-7}	30MB	No	S1	**Yes**

MEM1=Memory limits. MEM2=Memory management. S.CRT=Split criterion. S2=Gini index. RPA=Removing poor attributes.

5.3.2 Procedure

There will be a total of 15 parameter combinations for every dataset, indexed from A-O. Since there are a total of 4 datasets, the number of executions will be 60. Every execution represents the choice of applying one algorithm, with a specific parameter tunning, on one of the datasets. In parallel, we will be measuring how much energy is the execution consuming. Each combination of dataset and parameter tunning has been computed a total of ten times, to then obtain the average and standard deviation of all of them. The average of such computations are the results portrayed in the next section. A summary of these configurations is shown in Table 4. The experiment was carried out in a Linux machine with a 2.70 GHz Intel i7 processor (four cores), and with 8 GB of RAM. The models built from analyzing the synthetic generated data where trained and tested on 1 million instances. For the testing phase, new randomly generated data was used. On the other hand, for the real world datasets, the testing was performed on the same data as the training phase

Table 4 Design summary

	Quantity	Type
Datasets	4	Random Tree generator, Hyperplane generator, Poker, Airlines
Measures	6	Time, Power, Energy, Accuracy, Number of nodes, Tree depth
Parameter configuration	15	Represented in Table 3, as: A, B, C, D, E, F, G, H, I, J, K, L, M, N, O
Executions	10	10 executions for every parameter configuration on each dataset

5.3.3 Tools

We need to differentiate between two tools. The first tool, MOA (Massive Online Analysis), is used to execute the VFDT with the different parameter settings. Running in parallel to MOA is PowerAPI [21], a tool developed by the Spirals Research team [12], which is able to measure how much power different processes are consuming. PowerAPI [21] has been successfully tested by the authors [47] to compute the differences in terms of energy between some software process in different laptops. The energy is calculated by integrating the power consumed from the process during the execution time.

5.4 Evaluation

The last step of the experiment is the evaluation process. In order to evaluate the different settings on the different datasets, we have chosen four measures. The first two are accuracy and energy. We want to discover if there is a trade-off between energy and accuracy, i.e. we will only obtain a lower energy consumption by reducing the accuracy. Or, on the other hand, if there are specific setups were we can reduce energy consumption without loosing accuracy, i.e. smart setups. From the theoretical analysis on the algorithm we have observed that a possible relationship with accuracy is with the number of nodes of the tree. Therefore, the last two evaluation measures considered are the number of nodes (size) and the depth of the tree.

6 Results and Analysis

6.1 General analyses

Table 5 shows the results obtained from the experiment, for each dataset and measuring energy, time, power, number of nodes, depth of the tree and accuracy. First, we compare energy and accuracy, to understand if there is a visible trade-off between the increase of energy and the increase of accuracy. We can observe from Figure 1 how there seems to be a linear relationship between the increase of energy and the decrease of accuracy. Apparently, based on datasets 1, 3, and 4, whenever the energy increases the accuracy decreases. This result is promising in terms of our ultimate goal, trying to make energy efficient algorithms. Since there is no trade-off between energy and accuracy, a sacrifice of accuracy is not needed to develop energy efficient algorithms. In all datasets there is one outlier, that presents a lower accuracy than the rest, this is the parameter split criterion set to Gini index. This configuration presents a significantly lower levels of accuracy without reducing energy consumption in comparison with the other parameters.

Figure 2 shows how energy and accuracy vary for the different parameter configurations. Although this energy variation between parameters is analyzed in depth in the next subsection, we believe it is relevant to mention that energy differs significantly between specific configurations. For instance, for the third dataset, energy and accuracy vary for every single parameter, suggesting the relevancy of measuring energy consumption and not leaving the development choices to pure chance.

In this paragraph we analyze the relationship between the number of nodes and accuracy, portrayed in Figure 3. As has been explained in the theoretical analysis section, some parameter configuration can increase accuracy although intuitively it should be decreased. This is the case of the parameter τ. When τ is increased, intuitively the accuracy should decrease, since you are allowing splits on *not so good attributes*. However, although at first we predicted a decrease in accuracy, since there are significantly more nodes in the tree, the accuracy increases. This can be observed from Tables 5 and 6. The main reason is that with a significant increase of nodes, the tree is able to represent the data in a more fine-grained way, thus increasing accuracy. If we zoom in the highest values for each dataset, we can observe that for datasets 1, 3, and 4, there seems to be a higher accuracy whenever the number of nodes is higher.

A final analysis is related to the number of nodes and the energy consumed. When we look at the data from Table 5, whenever energy increases, time or power increases (since it is the product of both). The interesting measure is to see whether is the power or the time the one causing this energy increase. We have observed from

Table 5 Experimental results. The best accuracy and energy results for each dataset are highlighted.

			1						2			
S	T	P	E	A	N	D	T	P	E	A	N	D
A	4.44	8.59	38.10	96.91	1,134	8	5.85	7.39	43.19	90.93	655	12
B	3.90	7.32	28.57	96.31	661	8	5.65	7.12	40.22	90.99	607	11
C	3.90	7.10	27.71	96.24	570	7	5.66	7.75	43.90	91.22	575	10
D	3.82	6.99	**26.67**	95.91	495	7	5.57	7.77	43.31	90.98	515	11
E	4.46	8.44	37.63	95.57	699	8	5.36	9.15	49.08	90.57	57	7
F	4.74	7.35	34.84	97.93	1,541	9	7.42	6.89	51.10	90.38	2,071	19
G	5.09	7.30	37.11	97.94	2,074	11	9.25	6.87	63.55	89.87	3,863	18
H	5.20	7.55	39.21	**98.27**	2,181	12	9.51	6.78	64.45	89.75	3,971	19
I	4.58	7.90	36.22	97.54	1,403	9	6.32	7.01	44.32	90.82	1,129	12
J	4.29	8.18	35.10	96.35	888	8	5.73	7.58	43.48	**91.24**	471	11
K	3.98	7.37	29.31	95.43	1,134	8	5.03	7.60	**38.20**	84.89	655	12
L	4.40	8.32	36.60	96.91	1,134	8	5.92	7.17	42.48	90.93	655	12
M	4.39	7.95	34.91	96.91	1,134	8	5.91	7.12	42.11	90.93	655	12
N	6.12	6.67	40.79	83.11	1,735	117	5.77	7.60	43.90	90.72	619	11
O	4.34	7.85	34.08	96.91	1,134	8	5.83	7.24	42.23	90.93	655	12

			3						4			
S	T	P	E	A	N	D	T	P	E	A	N	D
A	7.29	7.33	53.48	76.63	297	16	6.54	9.17	59.96	67.31	8,582	4
B	7.03	8.29	58.27	70.67	181	11	6.09	8.41	51.20	67.01	7,984	3
C	7.25	8.57	62.16	79.44	195	16	6.06	8.40	50.91	67.01	8,228	4
D	7.70	8.76	67.45	73.66	149	16	5.99	8.19	**49.01**	66.65	7,895	3
E	8.36	7.49	62.60	73.74	149	13	6.19	8.81	54.50	67.09	5,127	3
F	7.14	7.46	53.26	82.56	791	17	6.94	8.46	58.74	67.77	12,307	3
G	7.46	7.53	56.25	84.88	1,285	22	7.32	8.41	61.56	**67.92**	14,137	3
H	6.87	7.06	**48.49**	**93.06**	1,991	20	7.29	8.43	61.46	67.90	14,001	4
I	7.32	7.36	53.88	78.93	575	18	6.75	8.61	58.14	67.70	10,871	3
J	7.89	8.98	70.89	68.71	119	13	6.41	9.14	58.58	66.97	6,618	3
K	7.26	7.43	53.92	76.63	297	16	6.56	8.96	58.74	67.31	8,582	4
L	7.19	7.57	54.41	76.63	297	16	6.51	8.88	57.82	67.31	8,582	4
M	7.34	7.61	55.85	76.63	297	16	6.60	9.47	62.48	67.31	8,582	4
N	8.99	7.25	65.17	37.83	1,380	102	7.62	9.24	70.37	58.77	1,490	53
O	7.18	7.37	52.95	76.63	297	16	6.52	9.06	59.07	67.31	8,582	4

T(s)=Time in seconds. P(W)=Power in Watts. E(J)=Energy in Joules. A(%)=Percentage of correctly classified instances. N=Nodes. D=Depth.

Figure 4 that whenever the number of nodes of a tree increase, so does the execution time. This phenomenon occurs more clearly for datasets 1, 2, and 4, suggesting that there is a relationship between time and nodes. Therefore, a conclusion is the increase of energy is related to an increase on time, probably due to an increase in the number of nodes.

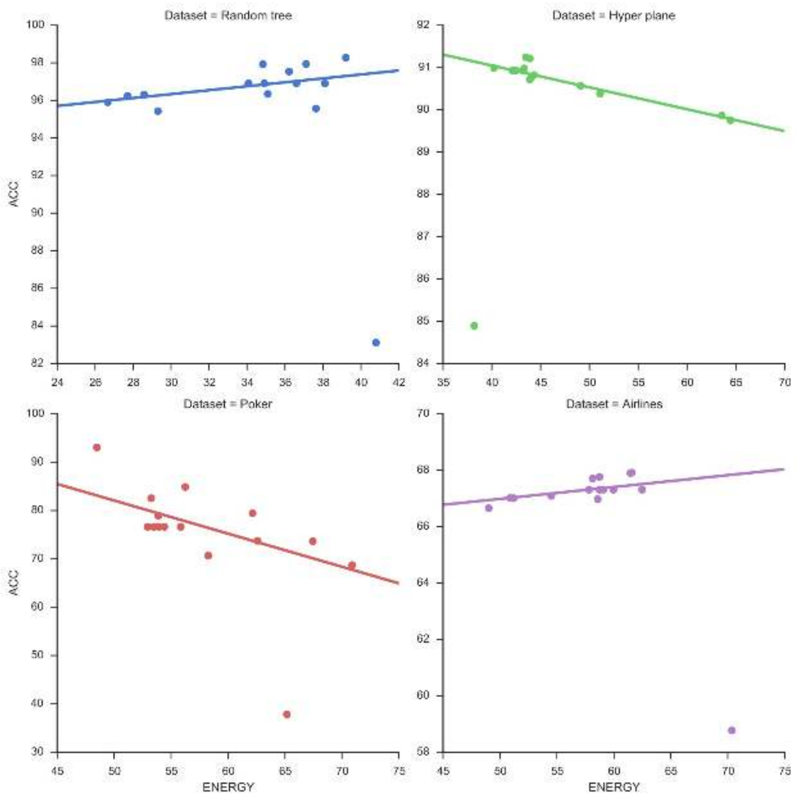

Fig. 1 Scatter plot with a linear relationship between accuracy (ACC) and energy (Joules) for every dataset.

A second measurement is power. We have observed that the power is not linearly correlated with the number of nodes. Figure 5 suggests that whenever the number of nodes is higher, the power is lower, higher or the same. We have not encountered a variable that is directly correlated with the increase of power, so we will analyze its behavior for each parameter in the next subsection.

6.2 Parameter analysis

This section aims to compare the parameters' behavior from the theoretical predictions with the empirical results. For that, we have created Table 6, that shows the summary of the real behavior of the parameters obtained in the experiment.

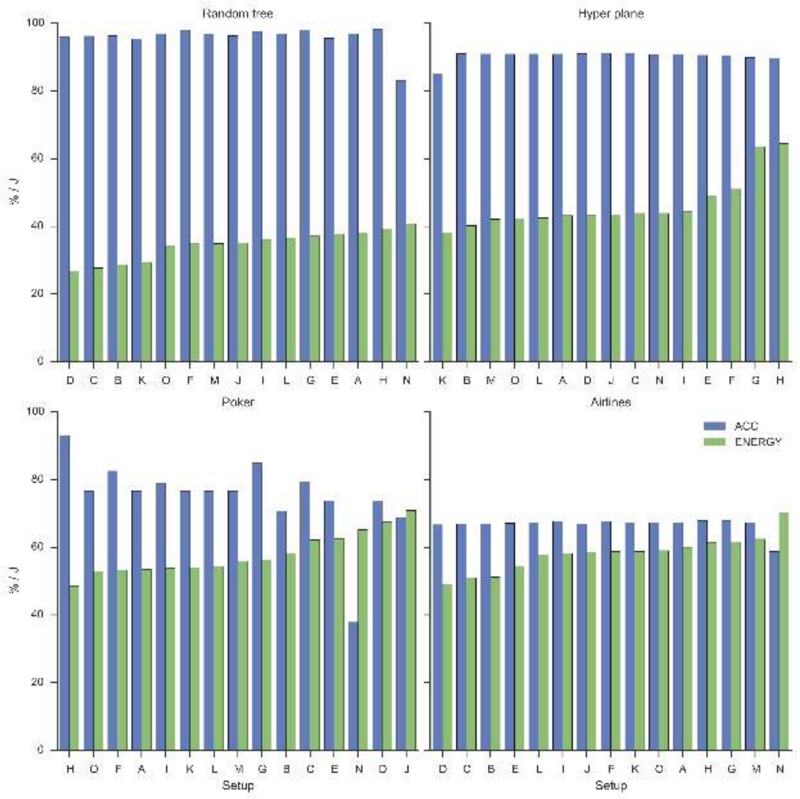

Fig. 2 Barplot showing the energy and accuracy variations for every parameter configuration on each dataset. Accuracy (ACC) is measured in percentage of correctly classified instances and energy is measured in Joules.

6.2.1 Parameter *nmin*

Observing the behavior of the *nmin* parameter across all datasets, we see that the accuracy is not significantly affected by this parameter, there is some decrease in dataset 1, but only of a 1%. In terms of power, however, we can see an important variation of Watts in datasets 1, 3, and 4. Both datasets 1 and 4 have an increase of power when *nmin* is increased. We checked what both datasets have in common to see the reasons behind this increase. The only characteristic that we find in common is that when power decreases, the depth of the tree is lower and also small in comparison to the other datasets. Another reason for this decrease in power when *nmin* increases is because we are computing less times the value of ΔG, therefore saving power. In terms of energy, when *nmin* increases, energy decreases and increases depending on the dataset. For datasets 1 and 4 it decreases, as was predicted in the theoretical section, for dataset 2 is stable and for dataset 3 it increases. In terms of nodes, we predicted that with higher *nmin* the number of nodes will decrease, and it

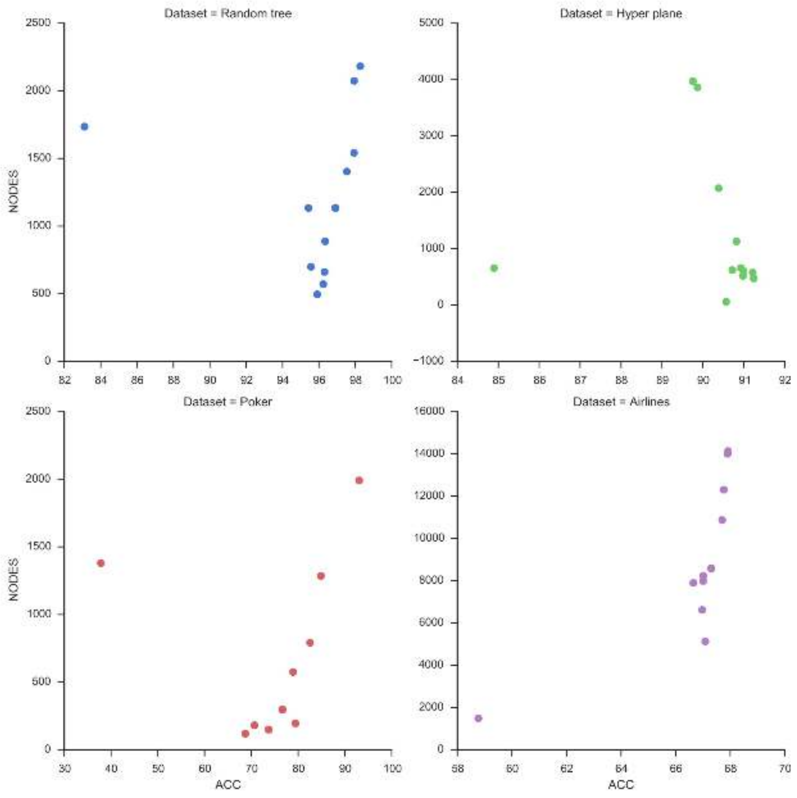

Fig. 3 Scatter plot with a linear relationship between accuracy and nodes for every dataset.

is exactly what happened in all datasets. Finally, looking at time, in general for all datasets except for the third one, time decreases when *nmin* increases, which is reasonable since we are looking into less batches of data, therefore the tree is computed faster.

6.2.2 Parameter τ

When the value of τ increases, so does the accuracy for datasets 1, 3, and 4. Dataset 2 experiences a non-stable accuracy value when increasing τ. Dataset 3 has an increase in accuracy of 11%, mainly due to the increase in the number of nodes, increasing around 2,000 nodes. That is also the case for all datasets, they significantly increase their number of nodes when τ increases, making this the main reason, from our understanding, for the accuracy increase. It matches with what we theoretically predicted, and with the results shown in the original paper of the authors [25]. In terms of energy, it varies depending on the parameter value. For datasets 2 and 4, there is a significant increase in energy. Dataset 3 experiences an important decrease

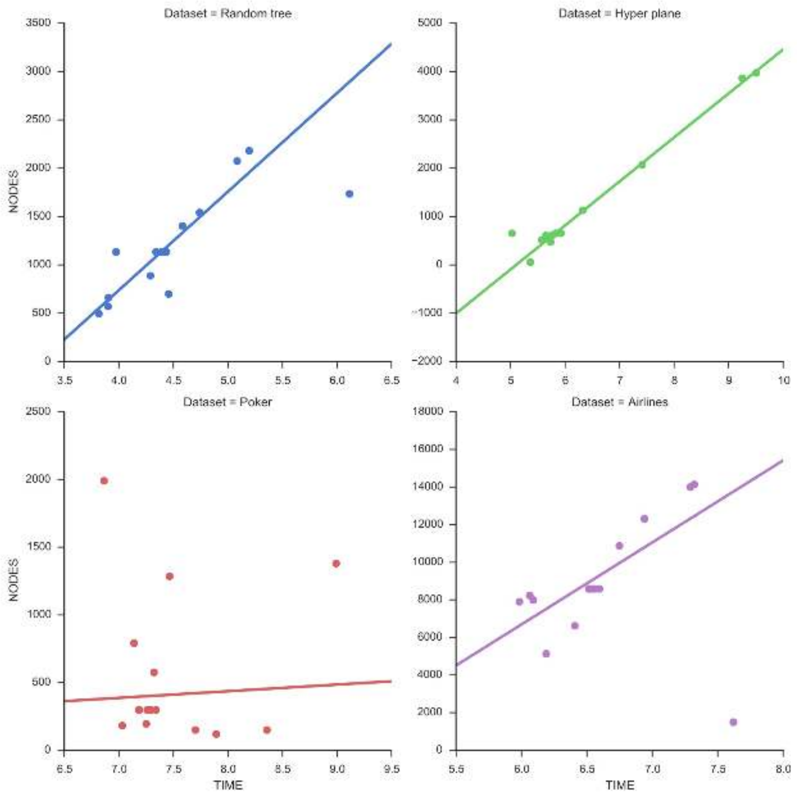

Fig. 4 Scatter plot with a linear relationship between time (seconds) and nodes for every dataset.

in energy and dataset 1 does not vary energy significantly. Datasets' 2 and 4 increase of energy is due to a significant increase in time, and a slight reduction in power. Maybe this increase in time is due to the time that it takes to build more nodes. Dataset 3 decreases energy because it decreases time, while power varies between the setups. We predicted that energy would increase, matching with the behavior of datasets 2 and 4.

6.2.3 Parameter δ

When δ decreases, the probability of making a correct split increases. In this case, datasets 1 and 3 experience a significant decrease in accuracy. Especially dataset 3, that varies its accuracy a 25% (from $\delta = 10^{-1}$ to $\delta = 10^{-10}$). Comparing δ from the default value to the highest value ($\delta = 10^{-1}$) , accuracy increases from 76.6% to 93%. Intuitively, it should be the opposite, that increasing δ would decrease accuracy, since there would be less confidence of making a correct split. However, since the number of nodes increases significantly, being almost 7 times more nodes for

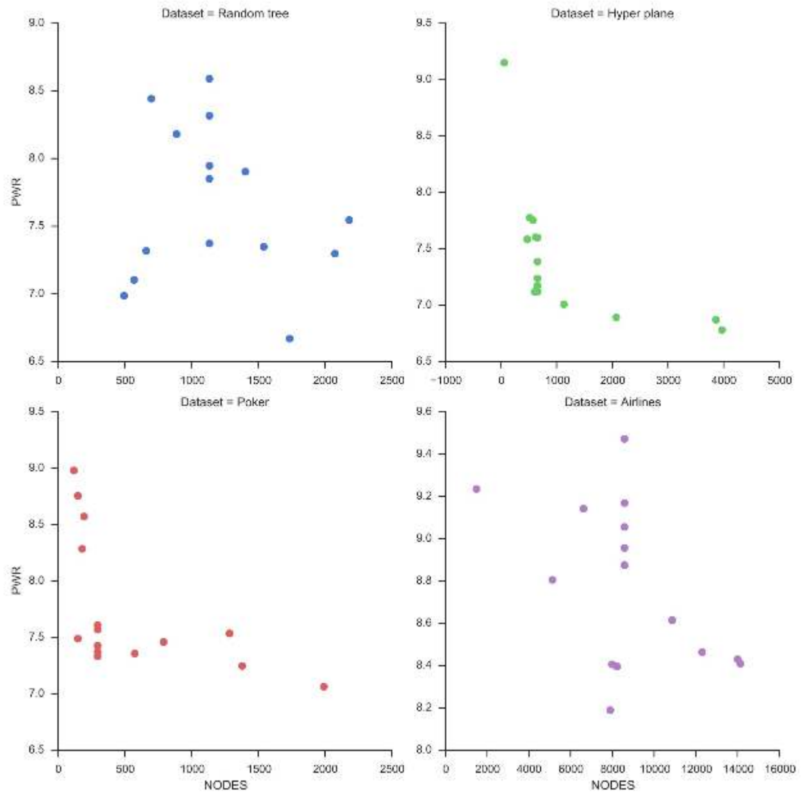

Fig. 5 Scatter plot with a linear relationship between power (W) and nodes for every dataset.

the value $\delta = 10^{-1}$, then accuracy increases by that much. In all datasets, when δ decreases, the number of nodes decreases also, as we predicted, since there are less splits. Taking a look into the energy aspect, when δ increases energy significantly decreases for datasets 1 and 2. Dataset 4 does not vary energy significantly and dataset 3 increases energy when δ decreases. We discovered that when $\delta = 10^{-1}$ for dataset 3, not only do we get a 25% increase in accuracy, as shown before, but we also get a decrease in energy of 22J.

6.2.4 Parameter MEM1

When the memory for the tree is restricted to 100KB, accuracy decreases by 1% for the first dataset and by 6% for the second dataset. For the other two datasets the accuracy stays the same, suggesting that the tree did not need more memory than 100KB. The number of nodes does not vary either in this case. The same happens when the memory is set to 2GB, the accuracy is the same across all datasets and the number of nodes does not vary at all. The reason the number of nodes is the same

Table 6 Summary of the real behavior of the parameters in the experiment. This table portrays how energy, accuracy and the number of nodes vary when increasing, decreasing or modifying certain parameters of the VFDT algorithm.

		nmin	τ	δ	MEM1		MEM2	S2	RPA
		↑	↑	↓	2GB	100KB			
D1	ACC	Dec	Inc	Dec	‡	Dec	‡	Dec	‡
	ENG	Dec	‡	Dec	Dec	Dec	Dec	Inc	Dec
	NDS	Few	More	Few	‡	‡	‡	More	‡
D2	ACC	‡	Dec†	Inc	‡	Dec	‡	‡	‡
	ENG	‡	Inc	Dec	Dec	Dec	Dec	Inc	Dec
	NDS	Few	More	Few	‡	‡	‡	Few	‡
D3	ACC	†	Inc	Dec	‡	‡	‡	Dec	‡
	ENG	Inc	Dec†	Inc	Inc	‡	Inc	Inc	Dec
	NDS	Few	More	Few	‡	‡	‡	More	‡
D4	ACC	‡	Inc	Dec	‡	‡	‡	Dec	‡
	ENG	Dec	Inc	Dec	Dec	‡	Inc	Inc	Dec
	NDS	Few†	More	Few	‡	‡	‡	Fewer	‡

†=It is not a constant decrease or increase throughout the configurations. Inc=Increase. Dec=Decrease. Few=Fewer nodes. More=More nodes. ACC=Accuracy. ENG=Energy. NDS=Nodes. ‡=There is no variation.

is because the implementation deactivates nodes when the memory is limited, but it does not remove them. In terms of power, time and energy, for the first two datasets there is a significant decrease of energy when the memory limit is set to 100KB. In the first dataset, the decrease is due to both a decrease in time and a decrease in power, which makes sense since the algorithm needs to analyze less number of nodes, consuming less power and taking less time. This is not the case for the two last datasets, since we already mentioned that probably the tree is not making use of such parameter, keeping energy at similar levels. When this parameter is set to the value of 2GB, the energy decreases slightly across all datasets except for the third one.

6.2.5 Parameter MEM2

When the parameter memory management (MEM2) is active, the tree stops growing when the memory limit is hit. The achieved accuracy and the number of nodes are exactly the same across all datasets. Energy does vary across all datasets, although it is not a big difference. Datasets 1 and 2 have a decrease in energy, while datasets 3 and 4 have an increase in energy. These variations are both due to the change of power and time, except for the second dataset, where there is a decrease on power but an increase on time.

6.2.6 Parameter S2

When the split criterion is set to *Gini index*, there is a significant decrease of accuracy across all datasets except for the second one. It is interesting to notice that even though it creates higher number of nodes for the first and third datasets, the accuracy is still significantly lower. In the second dataset the accuracy is maintained and the number of nodes decreased. In terms of energy, there is an increase of energy across all datasets, although power and time vary in a different way in all of them, in most of them when this parameter is chosen the time to build the tree increases. Based on these results, we would not recommend to choose this splitting criterion. However, we have not investigated the reasons behind these results, therefore we suggest to do that in future work studies.

6.2.7 Parameter RPA

When the parameter *removing poor attributes* is chosen, we expect to have a higher accuracy and a decrease in energy. From the experiment we can observe that accuracy is maintained across all datasets, and energy is decreased across also all datasets, although the decrease is not significant in all of them. This increase is mostly due to a slightly decrease in time and a decrease in power. The nodes are maintained for all datasets, which is not what we predicted. The reason could be that even though the tree is using less attributes, the amount of instances to analyze is the same.

6.2.8 Summary of the parameter analysis

In general, we can observe that tunning the parameters in a different way outputs different values of energy consumption and accuracy. At the same time, it depends on the type of dataset which parameters will give better results in terms of energy consumption. For instance, while for the first dataset, the set of *nmin* to 1,700 will give the best results, for the third dataset it gives very poor results in terms of energy consumption. The reason behind this is that the VFDT algorithm will create different models, different decision trees, depending on the input data. A general observation is that the third dataset, the poker dataset, behaves in a different way in comparison to the other three datasets. On the same line, setting *nmin* to 700 seems to give good results across all datasets except the third one, what suggests that is a good option for future researchers. From looking at Table 1, we conclude that, in general, *removing poor attributes* has a positive impact on energy consumption without affecting accuracy. Also, decreasing the value of δ, decreases the energy consumption and accuracy, but the accuracy decrease is not significant.

7 Conclusions and Future Work

The aim in this paper is to introduce energy consumption as an important factor during data mining algorithm evaluation and analysis. While performance and computational effort are factors usually considered in data mining, energy consumption is seldom evaluated. Energy awareness leads to reducing CO_2 emissions, increasing battery life of mobile devices and reducing air pollution.

In order to understand the impact of taking energy consumption into consideration, we have analyzed the behavior in terms of energy and accuracy of the VFDT (Very Fast Decision Tree) algorithm when modifying certain parameters. First, we theoretically analyzed how increasing or decreasing certain parameters of such algorithm would affect the tree structure, the accuracy and the energy consumed. Then, we created an experiment where we empirically vary the same parameters of the VFDT algorithm under four different datasets. We have compared the empirical with the theoretical results, and found that there is indeed a significant variation in terms of energy consumption that depends on how the algorithmic setup is designed. The results also indicate that it is possible to significantly reduce the energy consumption of an algorithm without reducing accuracy by varying correctly the parameters of the algorithm.

Future work is to investigate why certain parameter choices consume more energy than others. For this purpose, we aim to break down data stream mining algorithms into generic sub tasks to allow a more fine-grained comparison of energy consumption across various algorithms and algorithm configurations. Finally, we plan to obtain more challenging real world datasets to test how energy can vary on these type of datasets.

Acknowledgements This work is part of the research project ”Scalable resource-efficient systems for big data analytics” funded by the Knowledge Foundation (grant: 20140032) in Sweden.

References

1. Carbon footprint. `https://en.wikipedia.org/wiki/Carbon_footprint`
2. Digital universe study. `https://www.emc.com/collateral/analyst-reports/idc-digital-universe-united-states.pdf`
3. Ecological footprint. `https://en.wikipedia.org/wiki/Ecological_footprint`
4. Elena ikonomovska airline dataset. `http://kt.ijs.si/elena_ikonomovska/data.html`
5. Empowering developers to estimate app energy consumption. `http://research.microsoft.com/apps/pubs/default.aspx?id=166288`
6. Energy efficient software development. `https://software.intel.com/en-us/energy-efficient-software`
7. Google search statistics. `http://searchengineland.com/calculating-the-carbon-footprint-of-a-google-search-16105`
8. Green 500. `www.green500.org`

9. How much data is generated every minute on social media? `http://wersm.com/how-much-data-is-generated-every-minute-on-social-media/`. Published: August 19,2015
10. Poker dataset. `https://team.inria.fr/spirals/`
11. Search queries. `http://searchengineland.com/calculating-the-carbon-footprint-of-a-google-search-16105`
12. Spirals research group. `https://team.inria.fr/spirals/`
13. Address of the president, Lord Rees Of Ludlow om kt prs, given at the anniversary meeting on 1 december 2008. Notes and Records of the Royal Society of London **63**(2), pp. 183–190 (2009)
14. Aggarwal, C.C.: A framework for diagnosing changes in evolving data streams. In: Proceedings of the 2003 ACM SIGMOD international conference on Management of data, pp. 575–586. ACM (2003)
15. Aggarwal, C.C.: Data streams: models and algorithms, vol. 31. Springer Science & Business Media (2007)
16. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: On demand classification of data streams. In: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 503–508 (2004)
17. Agrawal, R., Imieliński, T., Swami, A.: Mining association rules between sets of items in large databases. In: ACM SIGMOD Record, vol. 22, pp. 207–216. ACM (1993)
18. Bhargava, R., Kargupta, H., Powers, M.: Energy consumption in data analysis for on-board and distributed applications. In: Proceedings of the ICML, vol. 3, p. 47 (2003)
19. Bifet, A., Holmes, G., Kirkby, R., Pfahringer, B.: MOA: massive online analysis. Journal of Machine Learning Research **11**, 1601–1604 (2010). URL `http://portal.acm.org/citation.cfm?id=1859903`
20. Bifet, A., Kirkby, R.: Data stream mining a practical approach (2009)
21. Bourdon, A., Noureddine, A., Rouvoy, R., Seinturier, L.: Powerapi: A software library to monitor the energy consumed at the processlevel. ERCIM News (2013)
22. Chu, S.: The energy problem and Lawrence Berkeley National Laboratory. Talk given to the California Air Resources Board (2008)
23. Datar, M., Gionis, A., Indyk, P., Motwani, R.: Maintaining stream statistics over sliding windows. SIAM Journal on Computing **31**(6), 1794–1813 (2002)
24. Ding, Q., Ding, Q., Perrizo, W.: Decision tree classification of spatial data streams using peano count trees. In: Proceedings of the 2002 ACM Symposium on Applied Computing, SAC '02, pp. 413–417. ACM, New York, NY, USA (2002)
25. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 71–80. ACM (2000)
26. Ferrer-Troyano, F., Aguilar-Ruiz, J.S., Riquelme, J.C.: Discovering decision rules from numerical data streams. In: Proceedings of the 2004 ACM symposium on Applied computing, pp. 649–653. ACM (2004)
27. Flinn, J., Satyanarayanan, M.: PowerScope: A tool for profiling the energy usage of mobile applications. In: WMCSA '99 Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications, pp. 2–10. IEEE (1999)
28. Gaber, M.M.: Data stream processing in sensor networks. In: Learning from Data Streams, pp. 41–48. Springer (2007)
29. Gaber, M.M., Krishnaswamy, S., Zaslavsky, A.: On-board mining of data streams in sensor networks. In: Advanced methods for knowledge discovery from complex data, pp. 307–335. Springer (2005)
30. Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: Mining data streams: a review. ACM Sigmod Record **34**(2), 18–26 (2005)
31. Gama, J.: Knowledge discovery from data streams. CRC Press (2010)
32. Gama, J., Gaber, M.M.: Learning from data streams. Springer (2007)
33. Gama, J., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. ACM Computing Surveys (CSUR) **46**(4), 44 (2014)

34. Garofalakis, M., Gehrke, J., Rastogi, R.: Querying and mining data streams: you only get one look a tutorial. In: SIGMOD Conference, p. 635 (2002)
35. Guha, S., Mishra, N., Motwani, R., O'Callaghan, L.: Clustering data streams. In: Proceedings 41st Annual Symposium on Foundations of Computer Science, pp. 359–366. IEEE (2000)
36. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American statistical association **58**(301), 13–30 (1963)
37. Hooper, A.: Green computing. Communication of the ACM **51**(10), 11–13 (2008)
38. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 97–106 (2001)
39. Jin, R., Agrawal, G.: Efficient decision tree construction on streaming data. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 571–576. ACM (2003)
40. Kearns, M.J., Vazirani, U.V.: An introduction to computational learning theory. MIT press (1994)
41. Kifer, D., Ben-David, S., Gehrke, J.: Detecting change in data streams. In: Proceedings of the Thirtieth international conference on Very large data bases-Volume 30, pp. 180–191 (2004)
42. Last, M.: Online classification of nonstationary data streams. Intelligent Data Analysis **6**(2), 129–147 (2002)
43. Law, Y.N., Zaniolo, C.: An adaptive nearest neighbor classification algorithm for data streams. In: Knowledge Discovery in Databases: PKDD 2005, pp. 108–120. Springer (2005)
44. Li, J., Martínez, J.F.: Power-performance considerations of parallel computing on chip multiprocessors. ACM Transactions on Architecture and Code Optimization (TACO) **2**(4), 397–422 (2005)
45. Martín, E.G., Lavesson, N., Grahn, H.: Energy efficiency in data stream mining. In: Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015, pp. 1125–1132. ACM (2015)
46. Naghavi, M., Wang, H., Lozano, R., Davis, A., Liang, X., Zhou, M., Vollset, S.E.V., Abbasoglu Ozgoren, A., Norman, R.E., Vos, T., et al.: Global, regional, and national agesex specific all-cause and cause-specific mortality for 240 causes of death: a systematic analysis for the global burden of disease study 2013. The Lancet **385**(9963), 117–171 (2015)
47. Noureddine, A., Rouvoy, R., Seinturier, L.: Monitoring energy hotspots in software. Automated Software Engineering **22**(3), 291–332 (2015)
48. Pillai, P., Shin, K.G.: Real-time dynamic voltage scaling for low-power embedded operating systems. In: ACM SIGOPS Operating Systems Review, vol. 35, pp. 89–102. ACM (2001)
49. Raileanu, L.E., Stoffel, K.: Theoretical comparison between the gini index and information gain criteria. Annals of Mathematics and Artificial Intelligence **41**(1), 77–93 (2004)
50. Rajaraman, A., Ullman, J.D.: Mining of massive datasets. Cambridge University Press (2011)
51. Reams, C.: Modelling energy efficiency for computation. Ph.D. thesis, University of Cambridge (2012)
52. Shafer, J., Agrawal, R., Mehta, M.: Sprint: A scalable parallel classifier for data mining. In: Proc. 1996 Int. Conf. Very Large Data Bases, pp. 544–555. Citeseer (1996)
53. Tiwari, V., Malik, S., Wolfe, A.: Power analysis of embedded software: a first step towards software power minimization. IEEE Transactions on Very Large Scale Integration (VLSI) Systems **2**(4), 437–445 (1994)
54. Utgoff, P.E.: Incremental induction of decision trees. Machine learning **4**(2), 161–186 (1989)
55. Wang, H., Fan, W., Yu, P.S., Han, J.: Mining concept-drifting data streams using ensemble classifiers. In: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 226–235. ACM (2003)
56. Zimmermann, H.: OSI reference model–the iso model of architecture for open systems interconnection. IEEE Transactions on Communications **28**(4), 425–432 (1980)