electronics

Article

Fast Characterization of Input-Output Behavior of Non-Charge-Based Logic Devices by Machine Learning

Arun Kaintura [1,*], Kyle Foss [1], Odysseas Zografos [2], Ivo Couckuyt [1], Adrien Vaysset [2], Tom Dhaene [1] and Bart Sorée [2]

1 IDLab, Department of Information Technology (INTEC), Ghent University—imec, Technologiepark—Zwijnaarde 15, 9052 Ghent, Belgium; Kyle.Foss@UGent.be (K.F.); Ivo.Couckuyt@ugent.be (I.C.); Tom.Dhaene@ugent.be (T.D.)
2 IMEC, Kapeldreef 75, B-3001 Leuven, Belgium; Odysseas.Zografos@imec.be (O.Z.); Adrien.Vaysset@imec.be (A.V.); bart.soree@imec.be (B.S.)
* Correspondence: arun.kaintura@gmail.com

Received: 17 June 2020; Accepted: 24 August 2020; Published: 26 August 2020

Abstract: Non-charge-based logic devices are promising candidates for the replacement of conventional complementary metal-oxide semiconductors (CMOS) devices. These devices utilize magnetic properties to store or process information making them power efficient. Traditionally, to fully characterize the input-output behavior of these devices a large number of micromagnetic simulations are required, which makes the process computationally expensive. Machine learning techniques have been shown to dramatically decrease the computational requirements of many complex problems. We use state-of-the-art data-efficient machine learning techniques to expedite the characterization of their behavior. Several intelligent sampling strategies are combined with machine learning (binary and multi-class) classification models. These techniques are applied to a magnetic logic device that utilizes direct exchange interaction between two distinct regions containing a bistable canted magnetization configuration. Three classifiers were developed with various adaptive sampling techniques in order to capture the input-output behavior of this device. By adopting an adaptive sampling strategy, it is shown that prediction accuracy can approach that of full grid sampling while using only a small training set of micromagnetic simulations. Comparing model predictions to a grid-based approach on two separate cases, the best performing machine learning model accurately predicts 99.92% of the dense test grid while utilizing only 2.36% of the training data respectively.

Keywords: machine learning; sequential sampling; data-efficient machine learning; magnetic logic devices; classification

1. Introduction

The scaling of conventional complementary metal-oxide semiconductors (CMOS) is reaching its limit [1] in accordance with Moore's prediction [2], introducing limitations and challenges to the semiconductor industry. As a result, various new concepts have emerged that aim to extend the semiconductor industry beyond CMOS technology [3,4]. Non-charge-based logic devices are one of the leading concepts [5] as these devices are power efficient and ultra-compact [6]. These devices can operate at high frequencies and offer new features such as non-volatility and low-voltage operation [5]. A number of such devices have been benchmarked in various publications for low-power applications [6–11].

With the need for solutions beyond CMOS, the research and development of novel non-charge-based logic devices have seen a great deal of interest in the past decade [3,4]. These logic devices rely on material properties to store information or perform logical operations. Nano-magnetic logic (NML), first introduced in [12,13] is a prominent concept in this category. This concept defines the state variable as magnetization direction (perpendicular magnetization) and information is processed through a dipolar coupling between nano-magnets. This allows computation to take place without passing any electric currents, making NML devices consume ultra-low power [6]. However, these devices possess certain limitations: Their operating frequency is restricted to about 3 MHz and the physical size to around 200 nm $\times$ 200 nm [14]. Contrary to the NML concept, a novel logic scheme was proposed in [15] based on the concept of bistable canted magnetization states. This scheme utilizes direct exchange interaction between two canted regions to perform logic operations and proves to be fast and power-efficient in comparison to other spin-based logic schemes [8,14].

The performance and ability of these devices to perform logic are dependent on various dynamics such as input field conditions and magnetization behavior. The key to characterizing the behavior of a new design is to identify input conditions for which the logic device behaves as desired. Traditionally, this is carried out by running a wide range of micromagnetic simulations (full grid sampling) using a simulator (such as Object-Oriented Micromagnetic Framework (OOMMF) or mumax [16,17]) for the micromagnetic system under study. The complexity of these devices increases with the number of logic structures, therefore, making simulations severely computationally expensive. Hence, there is a need to characterize these devices with minimal computational requirements.

Data-efficient machine learning (DEML) techniques have proven useful at reducing the computational requirements of a variety of engineering problems [18–21]. These techniques can be applied to micromagnetic problems for various objectives. One of the key applications of DEML in micromagnetics could be to characterize the behavior of a new design. The behavior of a device can be defined as operating if it performs a useful logical process and defined as not operating if it does not behave in a desired manner. In particular, in machine learning, such problems are identified as a classification problem that aims to separate a set of inputs into distinct groups (or classes). This is achieved by training a classifier or a model to a set of (training) data. Traditionally, this is a fixed data set. However, when data is expensive, the training data set can be generated via an adaptive sampling strategy. The adaptive algorithm starts with a small initial training data and adaptively enriches the training data by adding new samples from interesting regions in the design space. Henceforth, the trained classifier can be used to predict labels on any new unlabeled data. The main advantage of using DEML techniques over traditional practice is that full grid sampling is not required and only a small training set is sufficient to characterize device behavior. This significantly expedites the characterization of device behavior.

To apply novel DEML techniques to micromagnetic devices, we have considered a device that utilizes direct exchange interaction between two canted regions to perform logic operations [15]. Several state-of-the-art sampling strategies are combined with machine learning classification models. This paper evaluates the performance of the Explicit Design Space Decomposition (EDSD), Neighborhood-Voronoi (NV), Probability of Feasibility (PoF), and Entropy [18,22–24] sampling strategies. The performance of each technique is compared by three classifiers: Support Vector Machines (SVM), Gaussian process (GP), and Logistic regression (LR) [25–27] built on a training data that is obtained by adaptive sampling strategies. The preliminary analysis of the problem is presented in [28]. In the next section, various adaptive sampling algorithms and classification procedure are discussed.

2. Classification Methods

An adaptive sampling algorithm is used to intelligently select new training data in the input space in a sequential way. The adaptive sampling process can be model-dependent or model-independent depending on the sampling criteria or information utilized in the sampling process.

In the context of this work, we have used various adaptive sampling schemes that perform exploration and/or exploitation in the design space (NV, EDSD, PoF, and Entropy [18,22–24]). These techniques are discussed in the following subsections.

2.1. Neighborhood-Voronoi

Neighborhood-Voronoi is derived from the LOLA-Voronoi algorithm [29]. It is a model independent algorithm that requires no intermediate model construction during the selection of new training data. It has two components: Exploration (space-filling) and exploitation (refining boundaries), which are combined to identify boundaries of different class labels in the input space. NV maintains a balance between exploration and exploitation components, which allows the identification of previously undiscovered regions in the input space. One of the key advantages of using the NV algorithm is that no intermediate model is required during the selection of new training samples, which makes NV extremely efficient to execute. Given a set of K points, the exploration ensures that the input space is sampled as evenly as possible. A Voronoi-tessellation partitions the plane into C_k cells and the corresponding volume of each Voronoi cell is computed and assigned a score $V(x_k)$ (Equation (1)). Cells with larger relative volumes correspond to sparse areas and a higher score is assigned:

$$V_{\mathbf{x}_k} = \frac{Vol(C_k)}{Vol(C_1) + \cdots + Vol(C_K)} \;, \quad W(\mathbf{x}_k) = \begin{cases} 1 \Leftrightarrow \forall \; 1 \leq i,j \leq N : L(\mathbf{x}_k^i) = L(\mathbf{x}_k^j) \\ 0 \Leftrightarrow \exists \; 1 \leq i,j \leq N : L(\mathbf{x}_k^i) \neq L(\mathbf{x}_k^j) \end{cases} \tag{1}$$

$$G(\mathbf{x}) = V(\mathbf{x}) + W(\mathbf{x}) \quad \mathbf{x} \in \mathbf{t} \tag{2}$$

$$\mathbf{x}_{new} = \text{arg max min}||\mathbf{x} - (\mathbf{x}_k \cup \mathbf{t})|| \tag{3}$$

The NV-exploitation component refines the boundaries between different classes by favoring new samples in those regions. It computes a neighborhood of N points $\{\mathbf{x}_k^n\}_{n=1}^N$ for each chosen point $\mathbf{x}_k$. It should ensure that all neighbors are located closest to the chosen point while at the same time far apart from each other. Once the neighborhood is constructed, the labels of all neighbors $L(\mathbf{x}_k^n)$ are compared for any disagreement (mismatch). Any disagreement corresponds to the boundary region and a higher score $W(\mathbf{x}_k)$ is assigned to that Voronoi cell (Equation (1)). Finally both scores are combined $G(\mathbf{x})$ for each Voronoi cell and each cell is ranked (Equation (2)). The next sample location is then selected from the highest ranked cell. This is achieved by generating **t** random points in the ranked voronoi cell for each $\mathbf{x}_k$, and one point which is far away from other existing samples are chosen (Equation (3)).The process is repeated until the input region is sufficiently covered.

2.2. Explicit Design Space Decomposition

Explicit Design Space Decomposition [22] is a model-dependent technique that identifies boundaries between different classes. It requires intermediate models to be built during the selection of new samples. These models are explicitly used to define nonlinear boundaries or disjoint regions in the input space. These boundaries are treated as a limit state function/optimization constraint which is iteratively refined by adding new samples selected from regions where the misclassification probability is the highest. The reconstruction of classification boundaries continues until a converging criterion is met. Typically, EDSD uses Support Vector Machines (SVMs) to construct a limit state function. The SVM algorithm can efficiently handle discontinuities in the region. A SVM limit state function can be defined as:

$$s(x) = sign \left\{ b + \sum_{j=1}^{M} \lambda_i y_i K(x_i, x) \right\} \tag{4}$$

where b is a scalar quantity which is also noted as the bias and λ_i are the Lagrange multipliers. K is the SVM kernel function. Equation (9) can be used to classify any given arbitrary point in the design space depending on the positive or negative condition of s.

The selection of new training points begin by first generating initial training samples using Design of Experiments (DOE). In this work Centroidal Voronoi Tesellations (CVT) [30] is used to generate initial samples. The binary output to these observations is then calculated and a SVM limit state function is constructed. The SVM decision boundary is continuously refined by sampling new points on the SVM decision boundary that maximizes the distance to the nearest training sample. The process continues until the convergence criteria is met. The complete algorithm is shown in Figure 1.

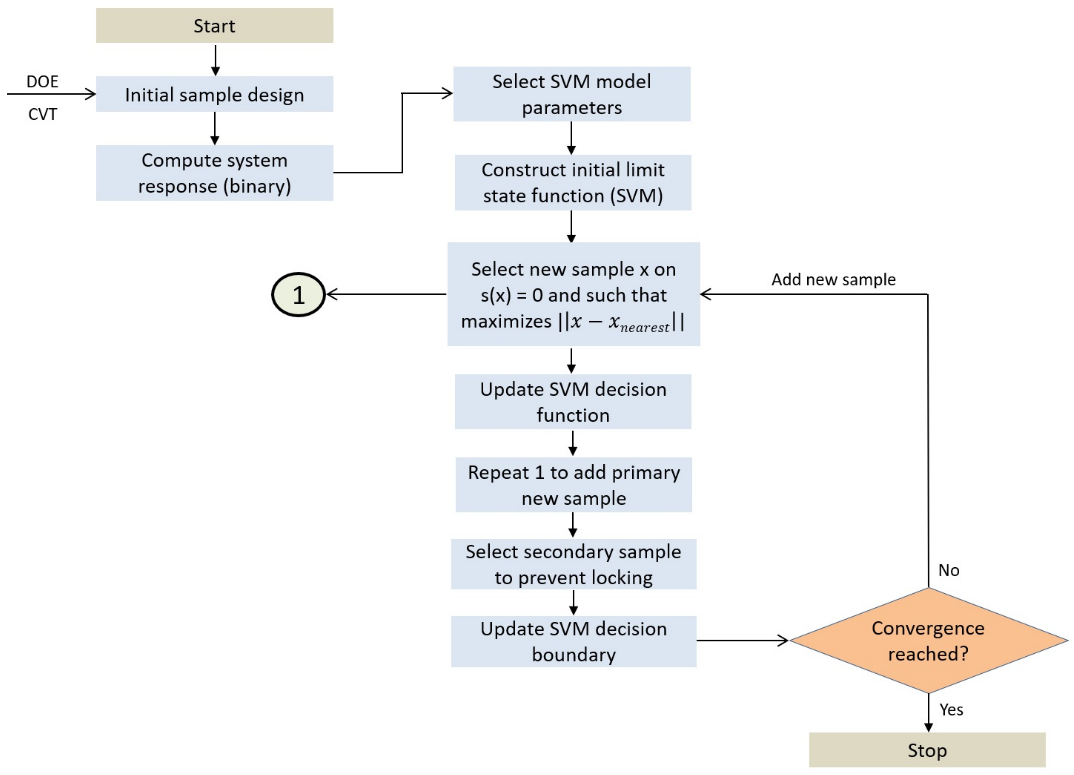

Start
DOE
CVT
Initial sample design
Compute system response (binary)
Select SVM model parameters
Construct initial limit state function (SVM)
1
Select new sample x on s(x) = 0 and such that maximizes $||x - x_{nearest}||$
Add new sample
Update SVM decision function
Repeat 1 to add primary new sample
Select secondary sample to prevent locking
No
Update SVM decision boundary
Convergence reached?
Yes
Stop

Figure 1. Process flow for model dependent on the Explicit Design Space Decomposition (EDSD) algorithm.

2.3. Probability of Feasibility

Probability of Feasibility is a model-based approach for adaptive design. In a classification problem [31], it gives a probabilistic estimate of a probabilistic classifier. The PoF criterion selects new samples in the design space that have a high probability of prediction to remain below a certain limit or threshold (g_{min}). In this work, this probability is multiplied by the candidate variance $\sigma^2_{\mathbf{x},\mathbf{D}}$ to include a component of exploration. Using the PoF criterion, any new point $\mathbf{x_{new}}$ can be selected as:

$$\mathbf{x}_{new} = \arg\max \left\{ \sigma^2_{\mathbf{x},\mathbf{D}} \left(P(F(\mathbf{x}) < g_{min}) \right) \right\} \tag{5}$$

$$= \arg\max \left\{ \sigma^2_{\mathbf{x},\mathbf{D}} \, \Phi \left(\frac{g_{min} - \mu_{\mathbf{x},\mathbf{D}}}{\sigma^2_{\mathbf{x},\mathbf{D}}} \right) \right\} \tag{6}$$

where Φ is the cumulative density function of the standard normal distribution. The PoF criterion is mostly used with the Gaussian process or Kriging models. Equation (5), $F(\mathbf{x})$ is a random variable with prediction mean $\mu_{\mathbf{x},\mathbf{D}}$ and variance $\sigma^2_{\mathbf{x},\mathbf{D}}$ at any point $\mathbf{x}$ and $\mathbf{D}$ in the observed data.

2.4. Entropy

Entropy is a model-dependent approach that can be interpreted as a measurement of homogeneity (or uncertainty) in the data. Using Bayesian Active Learning the information gain can be expressed in terms of predictive entropy, and parameter uncertainty can be minimized. Higher weight is given to samples that maximize the decrease in expected posterior entropy. In this work, we used Bayesian Active Learning by Disagreement [32] (BALD) algorithm and it computes entropies in the binary output space. Using BALD, the new point x_{new} that minimizes the entropy can be obtained as:

$$\mathbf{x}_{new} = arg \max_{\mathbf{x}} \; H[y|\mathbf{x}, \mathbf{D}] - E_{\theta \sim p(\theta|\mathbf{D})} \; [H[y|\mathbf{x}, \theta]] \tag{7}$$

$$= h\left(\Phi\left(\frac{\mu_{x,D}}{\sqrt{\sigma^2_{x.D} + 1}}\right)\right) - \frac{C}{\sqrt{\sigma^2_{x.D} + C^2}} \, exp\left(-\frac{\mu^2_{x,D}}{2\,(\sigma^2_{x.D} + C^2)}\right) \tag{8}$$

where θ is a latent parameter that controls the dependence between input $\mathbf{x}$ and output variables y, i.e., $p[y|\mathbf{x}, \theta]$ with p being the posterior distribution. The BALD algorithm requires posterior mean and variance to be computed. These posterior for each $\mathbf{x}$ can be easily computed using a GP model. For any $\mathbf{x}$, the objective in Equation (7) is simplified to Equation (8) for a GP model. $H[y|x, D]$ can be approximated to $h\left(\Phi\left(\frac{\mu_{x,D}}{\sqrt{\sigma^2_{x.D}+1}}\right)\right)$, C is $\left(\sqrt{\frac{\pi \log 2}{2}}\right)$, and $H[y|x, D] = h(\Phi(f(x)))$ can be expressed in terms of the entropy function h as $h(p) = -p \log p - (1 - p) \log(1 - p)$.

2.5. Classifier Description

In this section, various classifiers (SVM, GP, and LR) that are used in this work are briefly discussed. For a detailed discussion, interested readers are referred to [25–27]. Firstly, the SVM classifier can be given as:

$$s(x) = b + \sum_{j=1}^{M} \lambda_i y_i K(x_i, x) \tag{9}$$

where b is a scalar quantity, which is also noted the bias and λ_i are the Lagrange multipliers. K is the SVM kernel function. A suitable selection of the kernel function is vital for the performance of the SVM model. The Gaussian process model is widely used in regression problems owing to its well defined posterior formulation and computation. For classification problems, it is not possible to compute posterior quantities directly and a suitable approximation is required to compute posterior quantities. Classification models aim to predict the class label (y_i) for given test inputs (x_i). In a binary case, the probability to classify x_i in one of the two classes is given by:

$$p(y_i = 1|x_i, M) = \int p(y_i = 1|x_i, f) p(f|x_i, M) df \tag{10}$$

where M is the training set and f is a function to map. In many cases, the above expression is intractable and suitable approximation (such as Laplace approximation, expectation propagation) is required in order to obtain prediction mean and variance. Finally, in a binary case the probability to classify in one of the two classes using Logistic Regression is:

$$p(x_i) = \frac{exp(\beta_0 + \beta_1 x_i)}{1 + exp(\beta_0 + \beta_1 x_i)} \tag{11}$$

where the coefficients β_j can be obtained by maximum likelihood estimation.

In the next step, SVM, GP, and LR classifiers are trained on the training data collected by all adaptive algorithms. The constructed classifiers are validated against labeled test data to assess model performance. In the final step, the performance of all considered adaptive sampling algorithms is compared on different classifiers using various classification performance metrics [33]. The complete adaptive classification process is summarized in Figure 2. The initial samples are obtained by the Latin Hypercube Design (LHD [34]) and the corresponding output values (labels) are obtained by micromagnetic simulations. Next, a new batch of samples (sample) is obtained via an adaptive design algorithm. The output (labels) are computed for additional samples by micromagnetic simulations for mode M1 and by sub-sample from a dense grid sampling for mode M2. The process is repeated until the stopping criterion is reached. The classifier is then constructed on the final training data set. Note that in Figure 2, the highlighted portion and dotted area corresponds to the loop for model-dependent techniques only.

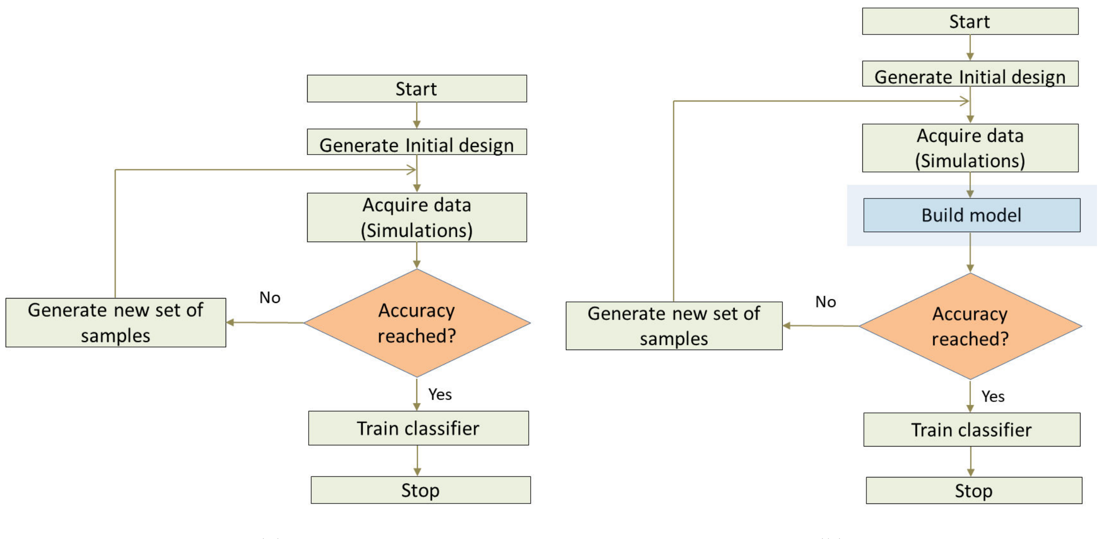


Figure 2. (**a**) Work flow for the model-independent process. (**b**) Work flow for the model-dependent process for adaptive sampling and classification.

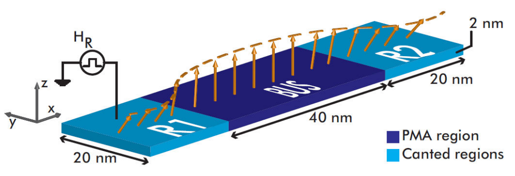
3. Logic Device Description

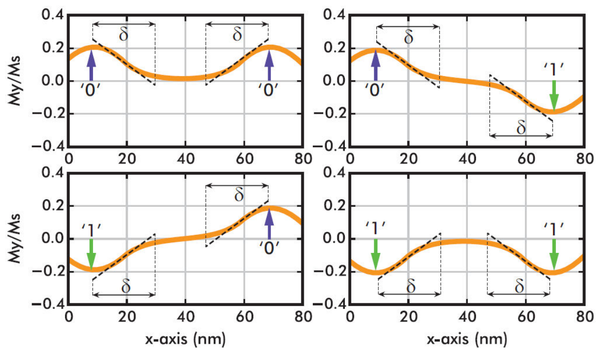
The structure of the logic device[15] evaluated is shown in Figure 3a. The device dimensions are 2 nm thick, 20 nm wide, and 80 nm long. It consists of two regions: Input (R1) and output (R2). R1 and R2 are interconnected through a magnetic bus and have in-plane and out-of-plane magnetic anisotropy along the $\hat{\mathbf{y}}$ direction respectively. To achieve a bistable canted magnetization, the length of the R1 and R2 regions are fixed as 20 nm. To avoid any strong exchange coupling between R1/R2, the interconnect length is set to 40 nm. Four possible combinations for the R1/R2 states can be defined based on the bistability of the canted magnetic regions. In the absence of an external magnetic field these states are defined in Figure 3b. The regions R1 and R2 can have a magnetization state '0' and '1' described by $M_y/M_s \cong 0.2$ and $M_y/M_s \cong -0.2$. The device is triggered by the application of an external magnetic field as shown in Figure 3a and a logic operation is performed. The applied magnetic field is parameterized by amplitude ($\mathbf{H_R}$) and duration ($\mathbf{T_R}$) and the behavior of the device responds according to these values. The external field will be applied at region R1 and it is desired to control the response of region R2. Thereafter, via magnetic exchange interaction, the logic state (0/1) of the region R1 is transmitted to region (R2).

In order for the device to perform logic, two possible logic operation modes M1 and M2 are defined, with both modes described in Table 1. For any 'XX' with $X \rightarrow (0,1)$ in Table 1 represent the logic state of the entire structure, for instance: '01' represent the logic state of the entire structure where '0' and '1' are the logic state of the region R1 and R2 respectively. Four possible stable states of the structure's magnetization are given as: '00', '01', '10', and '11' as shown in Figure (Figure 3b).

Table 1. Logic operation modes: 'XX' represents state of the region R1 and R2 where X is the unknown logic state of R1/R2 regions. The arrow represents the transition from the initial state of the structure to the final state.

Mode	**Logic Operation**
M1: State initialization	XX→X0 or XX→X1
M2: State propagation (BUF)	0X→X0 or 1X→X1
M2: State propagation (INV)	0X→X1 or 1X→X0





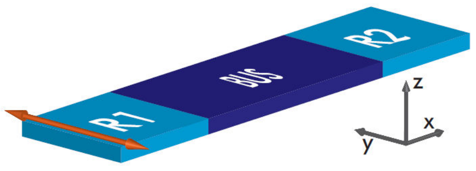
(**a**) (**b**)

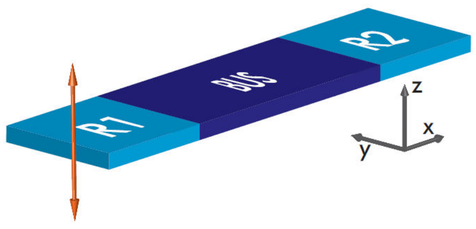
Figure 3. (**a**) The device under study: Input (R1) and output (R2) magnetic regions are interconnected through a ferromagnetic bus region of length 40 nm. [15] (**b**) Possible initial magnetization states of the structure with magnetization canted along the y-direction. BUF and INV corresponds to the buffer and inverter mode respectively.

Under the application of input field (triggering field) the magnetization dynamics of the input and output regions in the structure (see Figure 3a) change. The switching behavior is dependent on the magnitude, direction, and duration of the input field. For this purpose, we have considered two cases based on the direction of the applied input field, see Figure 4.

(a) Input field applied along the *y*-axis: corresponds to mode M1;
(b) Input field applied along the *z*-axis: corresponds to mode M2 (BUF and INV).

In this work, field amplitude and duration are parameterized in the domain: $0.5 \leq \mathbf{H_R} \leq 8$ (in kA/m) and $0.1 \leq \mathbf{T_R} \leq 0.5$ (in *ns*) respectively for input field application along the *y*-axis. In the case of field application along the negative *z*-axis the field amplitude is parameterized in the domain: $-8 \leq \mathbf{H_R} \leq -0.5$ (in kA/m) while $\mathbf{T_R}$ is the same as in case (a). The initial and final state of the regions R1/R2 corresponds to before and after the application of the external field. Based on the input and output states of the regions, it can be determined whether each region switched or not. Henceforth, from all input field conditions, interesting operating conditions can be extracted. For any triggered field the logic behavior of the structure in the mode M1/M2 can be extracted by micromagnetic simulations.





(**a**) (**b**)

Figure 4. (**a**) Field amplitude along ±y. (**b**) Field amplitude along ±z.

4. Results

The adaptive sampling process starts with initial samples, which are based on LHD. The corresponding labels are obtained by micromagnetic simulations using OOMMF. To assess the performance of all techniques a test set of 1271 samples is used. These samples are generated based on a full grid testing designed to sufficiently characterize device behavior. The results from adaptive sampling strategies are also compared with equivalent one-shot design generated by LHD on various classifiers. The performance of all considered approaches is assessed by employing the following classification performance metrics given for a binary case as:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

$$precision = \frac{TP}{TP + FP}, \quad recall = \frac{TP}{TP + FN} \tag{13}$$

where TP (true positives) is the number of positive class cases classified correctly, FP (false positives) is the number of negative class cases incorrectly classified as a positive class, FN (false negatives) is the number of positive class cases incorrectly classified as a negative class, ans TN (true negatives) is the number of negative class cases classified correctly. Precision defines exactness, i.e., what percentage of points that the classifier labeled as positive are actually positive. Recall measures what percentage of positive labels the classifier labeled as positive (best score is 1). The OOMMF micromagnetic solver [16] is used to perform all micromagnetic simulations whereas GPflowOpt [35], an open-source python-based package, is used to perform adaptive sampling based on PoF and Entropy criteria. The NV samples are generated using the SUMO toolbox [31,36]. To generate EDSD samples, the CODECS toolbox [37] is used.

4.1. Input Field Along Y-Axis

A positive external field is applied along the +*y*-axis as shown in Figure 4a. In the absence of any external field, the initial magnetization state of the structure is '01'. We are interested in the final state of R2 after the application of the external field. After field application, the final magnetization states of the structure will be either feasible ('00','10') or infeasible ('01','11'). This presents a binary classification problem where magnetization states '0' and '1' are represented by class 0 and class 1. The adaptive sampling algorithms initiates with an initial 5 samples obtained by a space-filling LHD design with an exception of EDSD where Centroidal Voronoi Tessellation (CVT) is used. The training data is extended by adding one sample at a time (adaptively) until a total training budget of 30 samples is reached.

The training samples generated by adaptive sampling techniques (EDSD, NV, PoF, and Entropy) and one-shot design (LHD) are plotted in Figure 5. For reference purposes, the true operating and no operation regions are also plotted in Figure 5a. While, EDSD performed (Figure 5c) more exploitation around the class boundaries, NV samples maintained a balance between exploration and exploitation (Figure 5d). In the case of Entropy and PoF, the samples obtained are well defined along the class boundary (Figure 5e,f). On the contrary, LHD results in a spaced filled design (Figure 5b and clearly neglects to sample around the class boundary. Overall, a good distribution of the obtained samples is observed for EDSD, NV, Entropy, and PoF techniques.

Table 2 summarizes the performance of all classifiers built on different training samples. In the case of PoF, 99.92% classification accuracy is achieved while one-shot design resulted in a 97.4% accuracy of the classifier. Note that, with a total number of 20 samples obtained adaptively (5 initial + 15 adaptively) by using the PoF algorithm, 99 percent classification accuracy is achieved on a GP classifier which is a significant improvement over accuracy obtained using a LHD design of 30 samples. Overall, NV, EDSD, Entropy, and PoF performed marginally better than the one-shot design. The sampling schemes EDSD, Entropy, and PoF sample well around the decision boundary which is highlighted in the higher precision and recall values. Moreover, EDSD performs exploitation uniformly around the class boundary.

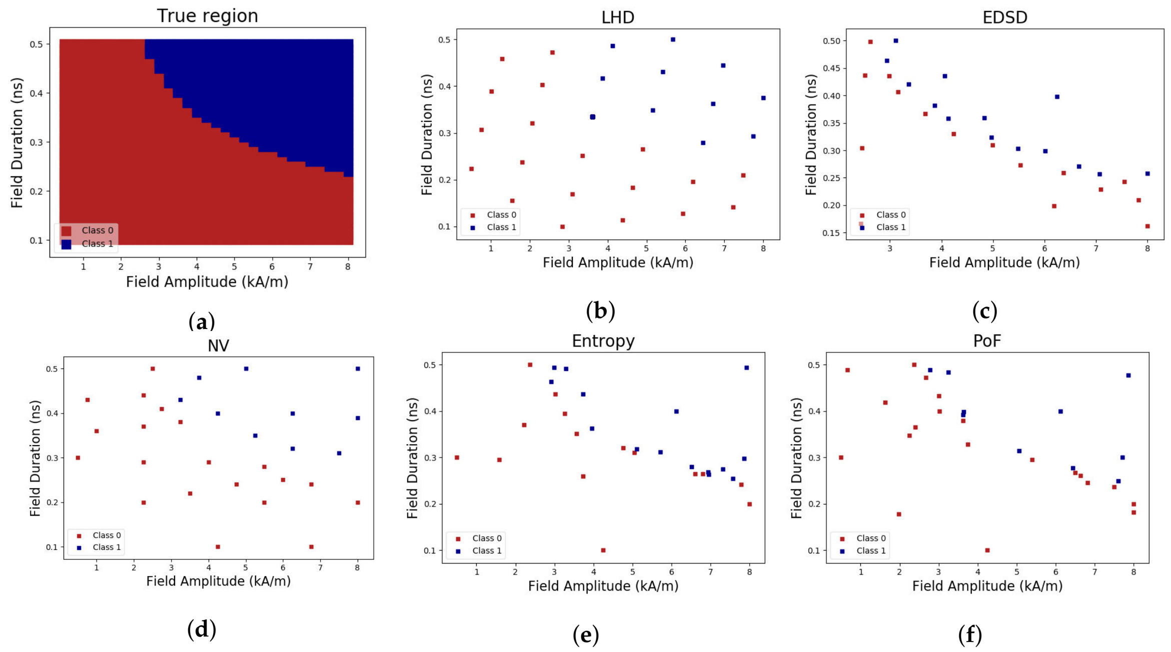


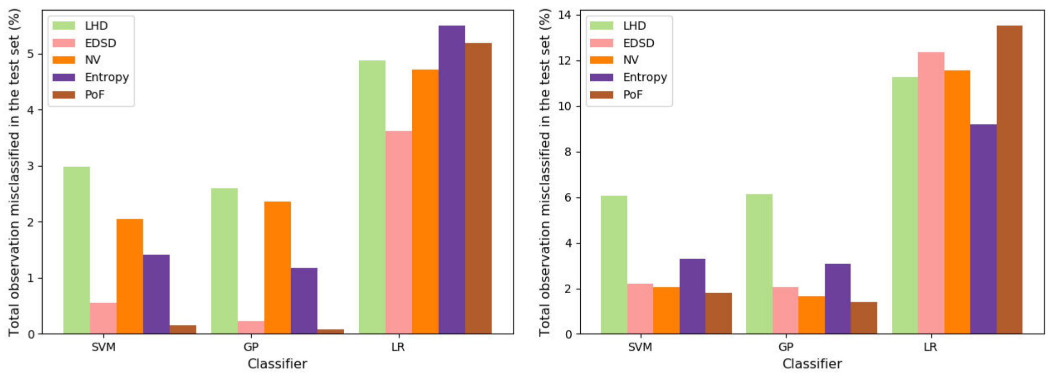
Figure 5. Sampling performed by Latin Hypercube Design (LHD), EDSD, Neighborhood-Voronoi (NV), Entropy and Probability of Feasibility (PoF) when external field is applied along + y-axis is shown in (**b**), (**c**), (**d**), (**e**) and (**f**) respectively. Red and blue squares represent points in the no operating (negative class) and operating regions (positive class, state 1) respectively. The complete operating and no operating regions are shown in (**a**).

Table 2. Case 1: Performance comparison of Support Vector Machines (SVM), Gaussian Process (GP), and Logistic Regression (LR) classifiers constructed on training data obtained from LHD, EDSD, NV, Entropy, and PoF. Mis. Obs. are the number of observations misclassified by the classifier for class 0 and 1. 'X' represents that no intermediate model is required in sampling technique. The intermediate models used in the model-dependent techniques are highlighted in the corresponding rows.

Algorithm	**Classifier**	**Model Dependent**	**Number of Samples**	**Test Set**				
				Precision	**Recall**	**Accuracy (%)**	**Mis. Obs.**	
							0	**1**
	SVM			0.97	0.97	97.00	27	11
LHD	GP	X	30	0.97	0.97	97.40	20	13
	LR			0.95	0.95	95.10	21	14
	SVM			0.99	0.99	99.44	2	5
EDSD	GP	SVM	5 + 25	1.0	1.0	99.76	1	2
	LR			0.96	0.96	95.59	22	24
	SVM			0.98	0.98	97.95	6	20
NV	GP	X	5 + 25	0.98	0.98	97.63	4	26
	LR			0.95	0.95	95.27	14	46
	SVM			0.99	0.99	98.58	2	16
Entropy	GP	GP	5 + 25	0.99	0.99	98.81	0	15
	LR			0.95	0.94	94.44	7	63
	SVM			1.0	1	99.84	0	2
PoF	GP	GP	5 + 25	1.0	1.0	99.92	0	1
	LR			0.95	0.95	94.80	9	57

Moreover, in Table 2 the number of misclassified observations by a classifier is compared for each class for different samplings. The results highlight how well a classifier can accurately predict labels around the class boundaries on a test set. Overall, GP and SVM classifier built on PoF samples results in the least misclassified observations in all classes.

The missclassification error (in percent), which is computed as the percent of total misclassified observations predicted by a classifier is visualized in Figure 6a for all cases. PoF and EDSD resulted in the least misclassified observations (0.078% and 0.236%) while LHD and NV (2.59% and 2.36%) have the highest missclassification error for the best scenario. The worst performance is reported by the LR classifier in all cases. Since LR is a linear model and the decision boundary is highly nonlinear, this performance of LR was expected.



(**a**) (**b**)

Figure 6. Total misclassified observations reported for each classifier type for test data (total percent): (**a**) External field along +y-axis and (**b**) external field along $-z$-axis.

4.2. Input Field Along Z-Axis

The case corresponds to the field application along the negative z-axis as shown in Figure 4b. Two state propagation behaviors are possible i.e., normal state propagation (buffer mode: BUF) and inverted state propagation (inverter mode: INV) corresponding to the transitions in mode $M2$ (Table 1). In the absence of any external field, the initial state of the structure is '01' and '00' for BUF and INV mode respectively. The feasible and infeasible final states for the BUF mode are ('00','10') and ('01','11') while for the INV mode are '01','11' and '00','10'.

This is a multi-class classification problem with class labels 0, 1, and 2 are assigned to NOOP (no operating), BUF, and INV modes respectively. This case is comparatively complex to classify as there exist regions in the input space which have a significantly small area (Figure 7a). The adaptive sampling initiates with 20 samples arranged in a TPLHD and a new sample is added one at a time until a total training data of 100 samples is reached.

The training samples obtained by running various sampling techniques are plotted in Figure 7. While for reference purpose, true BUF, INV, and NOOP regions are also plotted in Figure 7a. It is observed that in the case of Entropy and PoF, the samples are densely selected around the edges of the feasible regions (class boundaries). Moreover, with the same initial design for PoF, Entropy, and NV, both Entropy and NV are able to identify all operating/NOOP regions (fully/partially) while the PoF completely failed to identify narrow the BUF region. The EDSD algorithm is able to sample around all decision boundaries. However, it focused on local exploration around decision boundaries while missing global exploration (Figure 7c). The NV algorithm is able to perform global exploration and exploitation and results in sparser samples than EDSD. The performance of SVM, GP, and LR classifiers constructed on different training sets is reported in Table 3 for all cases. A significant improvement is observed in the classification accuracy of models constructed on adaptive samples over LHD. Overall, the best accuracy is achieved by a GP classifier with the PoF criteria. It shows an improvement of 4.72%, 5%, and 5% in accuracy, precision, and recall over the best performing classifier built on LHD respectively. Moreover, the number of misclassified observations predicted by a classifier for each class is also reported in Table 3.

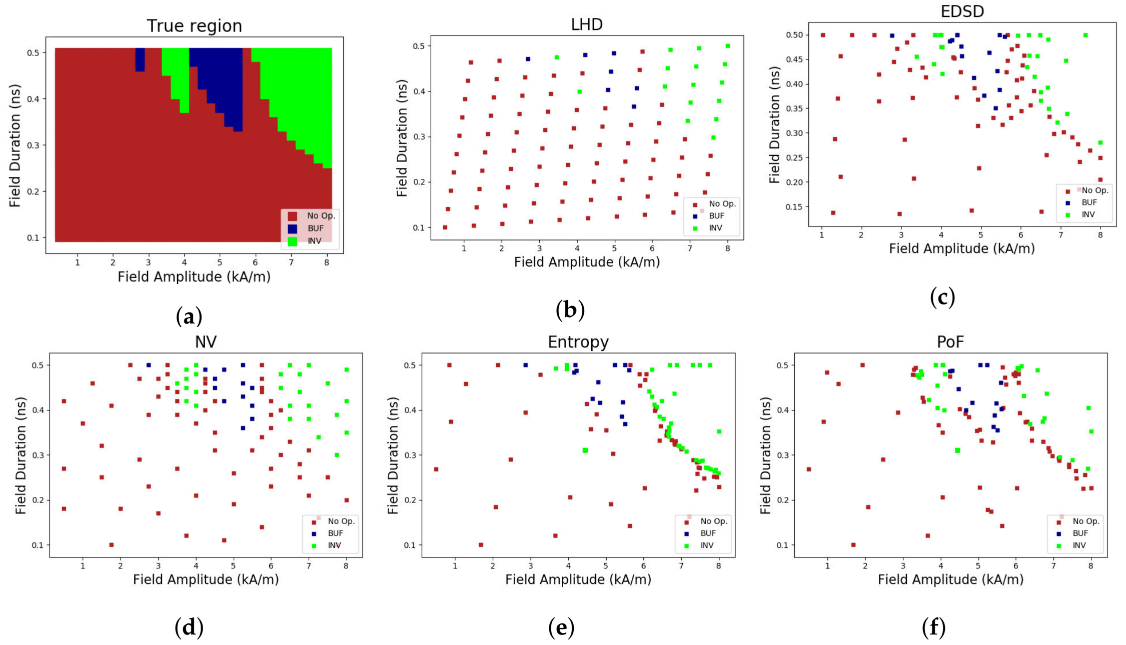


Figure 7. Sampling performed by LHD, EDSD, NV, Entropy, and PoF when external field is applied along the $-z$-axis is shown in (**b**), (**c**), (**d**), (**e**) and (**f**) respectively. Red, blue, and green squares represent points in: No operating, buffer, and inverter regions respectively. The complete operating, buffer, and inverter regions are shown in (**a**).

Table 3. Case 2: Performance comparison of GP classifier for all adaptive sampling and one-shot design. Mis. Obs. are the number of observations misclassified by the classifier for class 0, 1, and 2. 'X' represents that no intermediate model is required in the sampling technique. The intermediate models used in the model-dependent techniques are highlighted in the corresponding rows.

Algorithm	**Classifier**	**Model Dependent**	**Number of Samples**	**Test Set**					
				Precision	**Recall**	**Accuracy (%)**	**Mis. Obs.**		
							0	**1**	**2**
	SVM			0.94	0.93	93.15	53	11	23
LHD	GP	X	100	0.94	0.94	93.86	28	22	28
	LR			0.88	0.90	89.53	29	64	40
	SVM			0.98	0.98	97.79	11	8	9
EDSD	GP	SVM	20 + 80	0.98	0.98	97.95	4	11	11
	LR			0.82	0.88	87.64	15	72	70
	SVM			0.98	0.98	97.95	9	6	11
NV	GP	X	20 + 80	0.98	0.98	98.34	3	8	11
	LR			0.83	0.88	88.43	19	72	56
	SVM			0.97	0.97	96.69	17	5	20
Entropy	GP	GP	20 + 80	0.97	0.97	96.93	14	3	22
	LR			0.90	0.91	90.79	21	64	32
	SVM			0.98	0.98	98.19	16	4	3
PoF	GP	GP	20 + 80	0.99	0.99	98.58	3	7	8
	LR			0.81	0.86	86.46	13	72	87

The probability of missclassification is higher in the regions around the boundaries. The missclassification error also highlights how accurate boundaries are identified. In Figure 6b, the analysis of the total number of misclassified observations is performed for all classifiers. It can be seen that SVM and GP classifiers built on PoF, EDSD, and NV have similarly low missclassification errors, i.e., regions around the boundary are well identified. Moreover, the GP classifier utilizing PoF shows an improvement of 4.72% over a GP model built on LHD. The LR model performs poorly in all cases because of its linear behavior. These results are visualized in Figure 8 where true and learned class boundaries and misclassified observations are plotted for selected cases.

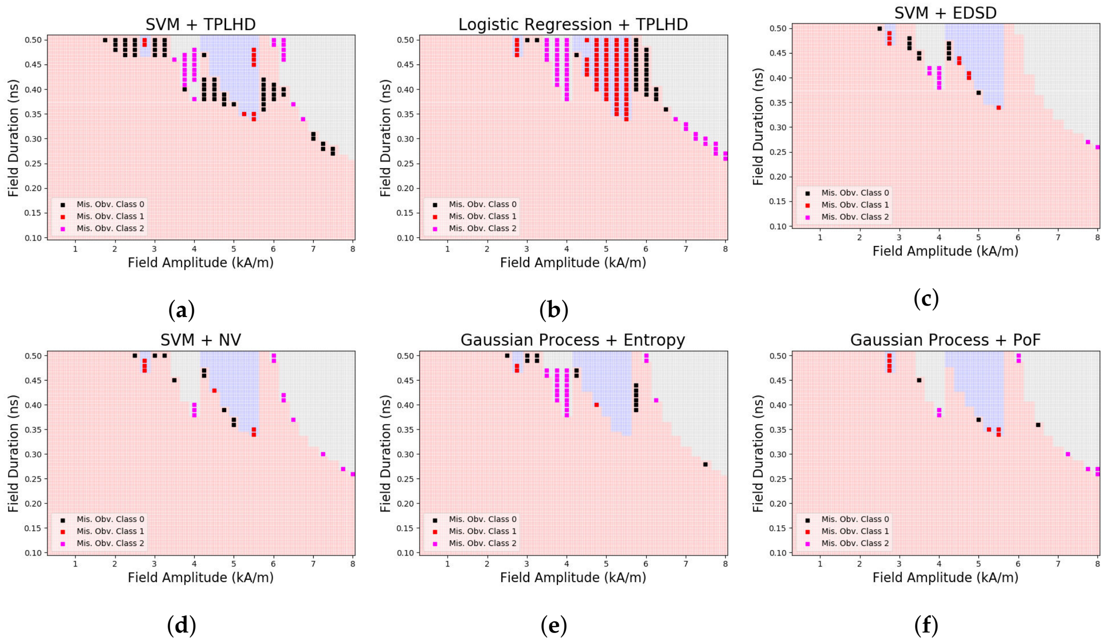


Figure 8. Misclassified observations by various classifiers which are trained on various sampling techniques: (**a**) SVM + TPLHD, (**b**) LR + TPLHD, (**c**) SVM + EDSD, (**d**) SVM + NV, (**e**) GP + Entropy and (**f**) GP + PoF.

5. Conclusions

The novel application of Data-Efficient Machine Learning techniques (DEML) is presented to characterize the behavior of non-charge-based logic devices. The adaptive sampling techniques substantially minimize the number of simulations (samples) required to characterize the dependence of input field conditions on the logic behavior. The performance of the various models and input-output-based adaptive sampling techniques are evaluated on classifiers built for binary and multi-class classification problems. The classification based on the adaptive sampling strategy significantly outperformed one-shot design and full grid sampling. In future work, the application of data-efficient machine learning techniques will be expanded to more challenging problems such as majority-based logic structures.

Author Contributions: A.K., I.C., T.D., K.F., O.Z., A.V., and B.S. developed the main idea. O.Z. provided OOMMF simulation model. A.K. performed analysis and wrote the manuscript. Funding acquisition, T.D., I.C., O.Z. and B.S. All authors reviewed the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Zhirnov, V.V.; Cavin, R.K.; Hutchby, J.A.; Bourianoff, G.I. Limits to binary logic switch scaling—A gedanken model. *Proc. IEEE* **2003**, *91*, 1934–1939. [CrossRef]
2. Moore, G.E. Cramming More Components Onto Integrated Circuits. *Proc. IEEE* **1998**, *86*, 82–85. [CrossRef]

3. Hutchby, J.A.; Bourianoff, G.I.; Zhirnov, V.V.; Brewer, J.E. Extending the road beyond CMOS. *IEEE Circuits Devices Mag.* **2002**, *18*, 28–41. [CrossRef]
4. Theis, T.N.; Wong, H.S.P. The End of Moore's Law: A New Beginning for Information Technology. *Comput. Sci. Eng.* **2017**, *19*, 41–50. [CrossRef]
5. Wolf, S.A.; Lu, J.; Stan, M.R.; Chen, E.; Treger, D.M. The Promise of Nanomagnetics and Spintronics for Future Logic and Universal Memory. *Proc. IEEE* **2011**, *98*, 2155–2168. [CrossRef]
6. Nikonov, D.E.; Young, I.A. Overview of Beyond-CMOS Devices and a Uniform Methodology for Their Benchmarking. *Proc. IEEE* **2013**, *101*, 2498–2533. [CrossRef]
7. Bernstein, K.; Cavin, R.K.; Porod, W.; Seabaugh, A.; Welser, J. Device and Architecture Outlook for Beyond CMOS Switches. *Proc. IEEE* **2010**, *98*, 2169–2184. [CrossRef]
8. Nikonov, D.; Bourianoff, G.I.; Ghani, T. Proposal of a Spin Torque Majority Gate Logic. *IEEE ELectron Device Lett.* **2011**, *32*, 1128–1130. [CrossRef]
9. Manfrini, M.; Kim, J.-V.; Petit-Watelot, S.; Roy, W.V.; Lagae, L.; Chappert, C.; Devolder, T. Propagation of magnetic vortices using nanocontacts as tunable attractors. *Nat. Nanotechnol.* **2014**, *9*, 121–125. [CrossRef]
10. Dutta, S.; Sou-Chi, C.; Nickvash, K.; Dmitri, N.; Manipatruni, S.; Young, I.A.; Naeemi, A. Non-volatile Clocked Spin Wave Interconnect for Beyond-CMOS Nanomagnet Pipelines. *Sci. Rep.* **2015**, *5*, 9861. [CrossRef]
11. Pan, C.; Naeemi, A. An Expanded Benchmarking of Beyond-CMOS Devices Based on Boolean and Neuromorphic Representative Circuits. *IEEE J. Explor. Solid State Comput. Devices Circuits* **2017**, *3*, 101–110. [CrossRef]
12. Cowburn, R.P.; Welland, M.E. Room Temperature Magnetic Quantum Cellular Automata. *Science* **2000**, *287*, 1466–1468. [CrossRef] [PubMed]
13. Csaba, G.; Imre, A.; Bernstein, G.H.; Porod, W.; Metlushko, V. Nanocomputing by field-coupled nanomagnets. *IEEE Trans. Nanotechnol.* **2002**, *1*, 209–213. [CrossRef]
14. Breitkreutz, S.; Kiermaier, J.; Eichwald, I.; Hildbrand, C.; Csaba, G.; Schmitt-Landsiedel, D.; Becherer, M. Experimental Demonstration of a 1-Bit Full Adder in Perpendicular Nanomagnetic Logic. *IEEE Trans. Magn.* **2013**, *49*, 4464–4467. [CrossRef]
15. Zografos, O.; Manfrini, M.; Vaysset, A.; Sorée, B.; Ciubotaru, F.; Adelmann, C.; Lauwereins, R.; Raghavan, P.; Iuliana, P.R. Exchange-driven Magnetic Logic. *Sci. Rep.* **2017**, *7*, 12154. [CrossRef] [PubMed]
16. Donahue, M.; Porter, D. OOMMF User's Guide, Version 1.0. 1999. Available online: http://math.nist.gov/oommf (accessed on 15 January 2019).
17. Vansteenkiste, A.; Leliaert, J.; Dvornik, M.; Helsen, M.; Garcia-Sanchez, F.; Waeyenberge, B.V. The design and verification of MuMax3. *AIP Adv.* **2014**, *4*, 107133. [CrossRef]
18. Singh, P.; Herten, J.V.D.; Deschrijver, D.; Couckuyt, I.; Dhaene, T. A sequential sampling strategy for adaptive classification of computationally expensive data. *Struct. Multidiscip. Optim.* **2017**, *55*, 1425–1438. [CrossRef]
19. Omar, Y.A.J.; Paul, D.Y.; Muhaidat, S.; Karagiannidis, G.K.; Taha, K. Efficient Machine Learning for Big Data: A Review. *Big Data Res.* **2015**, *2*, 87–93.
20. Singh, P.; Deschrijver, D.; Pissoort, D.; Dhaene, T. Adaptive classification algorithm for EMC-compliance testing of electronic devices. *Electron. Lett.* **2013**, *49*, 1526–1528. [CrossRef]
21. Basudhar, A.; Dribusch, C.; Lacaze, S.; Missoum, S. Constrained efficient global optimization with support vector machines. *Struct. Multidiscip. Optim.* **2012**, *46*, 201–221. [CrossRef]
22. Basudhar, A.; Missoum, S. An improved adaptive sampling scheme for the construction of explicit boundaries. *Struct. Multidiscip. Optim.* **2010**, *42*, 517–529. [CrossRef]
23. Alexander, I.J.F.; Andy, J.K. Recent advances in surrogate-based optimization. *Prog. Aerosp. Sci.* **2009**, *45*, 50–79.
24. Shannon, C.E. A Mathematical Theory of Communication. *Assoc. Comput. Mach.* **2001**, *5*, 3–55. [CrossRef]
25. Cortes, C.; Vapnik, V. Support-vector networks. *Mach. Learn.* **1995**, *20*, 273–297. [CrossRef]
26. Rasmussen, C.E.; Williams, C.K.I. *Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning)*; The MIT Press: Cambridge, MA, USA, 2005.
27. Lee, W.S.; Liu, B. Learning with Positive and Unlabeled Examples Using Weighted Logistic Regression. In Proceedings of the Twentieth International Conference on International Conference on Machine Learning, ICML'03, Washington, DC, USA, 21–24 August 2003; AAAI Press: Palo Alto, CA, USA, 2003; pp. 448–455.

28. Kaintura, A.; Foss, K.; Couckuyt, I.; Dhaene, T.; Zografos, O.; Vaysset, A.; Sorée, B. Machine Learning for Fast Characterization of Magnetic Logic Devices. In Proceedings of the 2018 IEEE Electrical Design of Advanced Packaging and Systems Symposium (EDAPS), Chandigarh, India, 16–18 December 2018; pp. 1–3.
29. Crombecq, K.; Gorissen, D.; Deschrijver, D.; Dhaene, T. A Novel Hybrid Sequential Design Strategy for Global Surrogate Modeling of Computer Experiments. *SIAM J. Sci. Comput.* **2011**, *33*, 1948–1974. [CrossRef]
30. Romero, V.J.; Burkardt, J.V.; Gunzburger, M.D.; Peterson, J.S. Comparison of pure and Latinized centroidal Voronoi tessellation against various other statistical sampling methods. *Reliab. Eng. Syst. Saf.* **2006**, *91*, 1266–1280. [CrossRef]
31. Herten, J.V.D.; Couckuyt, I.; Deschrijver, D.; Dhaene, T. Adaptive classification under computational budget constraints using sequential data gathering. *Adv. Eng. Softw.* **2016**, *99*, 137–146. [CrossRef]
32. Houlsby, N.; Huszár, F.; Ghahramani, Z.; Lengyel, M. Bayesian Active Learning for Classification and Preference Learning. *arXiv* **2011**, arXiv:1112.5745.
33. Forman, G. An Extensive Empirical Study of Feature Selection Metrics for Text Classification. *J. Mach. Learn. Res.* **2003**, *3*, 1289–1305.
34. Felipe, A.C.V.; Venter, G.; Balabanov, V. An algorithm for fast optimal Latin hypercube design of experiments. *Int. J. Numer. Methods Eng.* **2010**, *82*, 135–156.
35. Knudde, N.; Herten, J.V.D.; Dhaene, T.; Couckuyt, I. GPflowOpt: A Bayesian Optimization Library using TensorFlow. *arXiv* **2017**, arXiv:1711.03845.
36. Gorissen, D.; Couckuyt, I.; Demeester, P.; Dhaene, T.; Crombecq, K. A Surrogate Modeling and Adaptive Sampling Toolbox for Computer Based Design. *J. Mach. Learn. Res.* **2010**, *11*, 2051–2055.
37. Sylvain, L.; Missoum, S. CODES: A Toolbox For Computational Design Version 1.0. 2015. Available online: www.codes.arizona.edu/toolbox (accessed on 15 January 2019).