Energy-Efficient Data Symbol Detection via Boosted Learning for Multi-Actuator Data Storage Systems

Jiachen Xu, Ethan Chen, and Vanessa Chen
Department of Electrical and Computer Engineering
Carnegie Mellon University, Pittsburgh, PA, USA
Email: {jiachen, ethanchen, vanessachen}@cmu.edu

2021 IEEE International Symposium on Circuits and Systems (ISCAS) | 978-1-7281-9201-7/20/$31.00 ©2021 IEEE | DOI: 10.1109/ISCAS51556.2021.9401676

***Abstract*—Machine-learning-based readout channels are presented for direct data symbol detection via decision-tree classification with gradient boosting for multiple-actuator data storage systems. The proposed learning module integrates energy-efficient linear classifiers to extract features and structures from raw readback signals. The results demonstrate high detection accuracy, which is robust to inter-symbol interference (ISI) and jitter noise. The low-complexity machine learning module classifies low signal-to-noise ratio raw data with an accuracy rate higher than 95.8% in real-time and consumes only 53 mW.**

Keywords—machine learning, FPGA, classification, data detection

I. INTRODUCTION

Data centers that can offer low-latency data access, reliable data storage and high-performance computing are key enablers for a densely connected world. To provide the platform for multitask processing, multiple readout channels that parallelly process complex signals from hard disk drives (HDDs) for multiple-actuators are exploited to improve the access efficiency and computation performance. Meanwhile, heat-assisted magnetic recording (HAMR) [1] has proved to be a promising technology for future magnetic data storage products with high areal density [2][3]. During the write process of HAMR, near-field plasmonics technology is utilized to elevate the media temperature close to the Curie point. However, HAMR-specific challenges, such as a switching field distribution that is unique to HAMR recording and leads to additional transition jitter and saturation noise [4], are required to be addressed in the backend. To mitigate noise issues and inter-symbol interference (ISI) due to high-speed interfacing, a conventional digital backend consists of equalizers, whitening noise filters, and decoders to carry out data detection algorithms, such as Partial-Response Maximum-Likelihood (PRML) and Noise-Predictive Maximum-Likelihood (NPML) [5]. In recent studies, high-dimensional machine learning (ML) algorithms [6] may offer alternatives for error reduction, but the computational complexity is not suitable for energy-efficient hardware implementation of parallel channels in a multiple-actuator platform. Typical communication signal analysis involves pre-processing to diminish dimensionality so that high-order signals can be removed from the set of points in the trace. Nevertheless, it is still computationally expensive to realize pre-processing and feature-extraction algorithms in parallel read channels to mitigate the influence of un-predictable jitter and noise from HAMR recording.

Decision trees (DTs) are one of the most efficient algorithms used in ML for classification, but its error rate increases when feature distribution is not a simple 2-plane case. Process variations and noise influence can result in a significant degradation of classification accuracy. The weak learner like DTs can be improved by using ensembles that combine a group of linear learners to form a strong learner. The Gradient Boosted Decision Trees (GBDTs) is deployed to break the trade-offs between accuracy and hardware complexity [7]. To eliminate the need of pre-processing the raw data, energy-efficient decision tree algorithms with accuracy-boosting methods are exploited to enable strong classification with linear classifiers to perform direct data detection from raw signals. Specifically, the boosted decision tree interface (BDTI) exploits a data-driven approach to detect complex datasets, so that it can rapidly learn noise behavior and ISI and adapt to characteristics of magnetic media, heads, and detection channels. In this way, the ML data detector can offer better generalization and adaptability to enable an energy-efficient multitasking data-storage system.

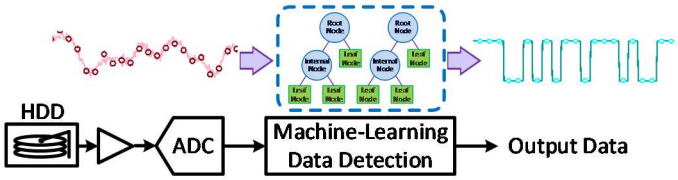


Fig. 1. Machine-learning-based readout channels for multiple-actuator data storage systems: the boosted decision trees learn features of magnetic medium and recording heads to characterized data-clock behavior for error-aware data detection.

II. THE SYSTEM ARCHITECTURE

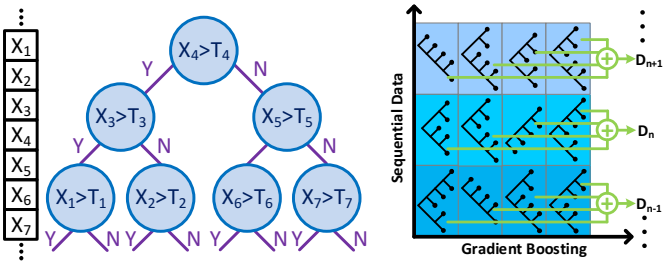
The proposed computational readout channel is shown in Fig. 1, which consists a low-noise amplifier, an analog-to-digital converter (ADC) and the ML data classification module. The ML module directly processed quantized raw data from a 6-bit ADC for data detection without digital equalization and error correction code (ECC). Among those nonidealities of read channels and the magnetic medium, correlated noise and nonlinearity, such as jitter noise and ISI, are main factors that dominate bit error rates (BERs), which are significantly improved with the proposed low-complexity BDTI as the ML module can efficiently learn correlated data symbol structures from the noisy and distorted signals.

Fig. 2(a) shows the hierarchy of the DTs. The principle of the DTs is that the object is first categorized to a certain group with a specific feature. Decision making is iterative until the object arrives a leaf. The arrangement of a group in such decision-making processes forms a DT, where each leaf

This work was supported in part by the National Science Foundation CARRER program under Grant No. 1953801 and the Data Storage Systems Center (DSSC) at Carnegie Mellon University.

978-1-7281-9201-7/21/$31.00 ©2021 IEEE

represents one category. At each node the set is divided again based on its features. Gradient boosting is exploited to enhance classification accuracy for fast data detection with good energy efficiency. As shown in Fig. 2 (b), the learner is trained iteratively through the pseudo residuals of the learner. The computing for pseudo residuals of the DTs is shown in Algorithm 1. In this paper feature decisions and gradient calculations are realized in low-complexity FPGA hardware to perform high-accuracy classification for real-time data symbol detection from the noisy and distorted signals.



(a) (b)

Fig. 2. (a) The decision tree that utilizes multiple samples as the features and (b) the gradient boosting method to improve the accuracy.

Algorithm 1: Gradient Boosting

- Initialize $F_0(x) = \arg\min_{\rho} \sum_{i=1}^{N} L(y_i, \rho)$
- For m = 1 to M:
  1. Compute the pseudo-residuals:
  $$\tilde{y}_i = -\left[\frac{\partial L(y_i, F(x_i))}{\partial F(x_i)}\right]_{F(x)=F_{m-1}(x)} \quad for\ i = 1, \ldots, n.$$
  2. Fit a base learner (tree) $h_m(x)$ to pseudo-residuals using the training set $\{(x_i, \tilde{y}_i)\}_{i=1}^{n}$
  3. Choose a gradient descent step size as
  $$\rho_m = \arg\min_{\rho} \sum_{i=1}^{N} L\left(y_i, F_{m-1}(x_i) + \rho h_m(x_i)\right)$$
  4. Update the model:
  $$F_m(x) = F_{m-1}(x) + \rho_m h(x)$$
  end For
- Output the final regression function $F_m(x)$

III. Algorithm Realization

Fig. 3 shows the structure of a DT and its inherent memory buffers that contain respective parameters for this design. Each node in the DT has an index and the root node's index is 0. The threshold buffer stores the value that is going to be compared with the input features to make a decision on the nodes. The comparator buffer contains the index of the input feature for non-leaf node to be compared with. The score buffer defines each leaf node's score. The left children buffer and the parent buffer are structure buffers that describe the structure of the binary tree for traversing the tree when making the decisions. In this design all of the DTs in the GBDT ensemble are complete binary trees with identical structures. For a complete binary tree with depth h, the tree has $2^{h+1} - 1$ nodes, and 2^h of them are leaf nodes. The binary trees with identical structures can use the same structure buffers, which eliminates the need for storing different structure buffers for each tree. In the FPGA hardware design, all of the trees share the same structure memory to traverse the nodes, which reduces the usage of FPGA memory significantly.

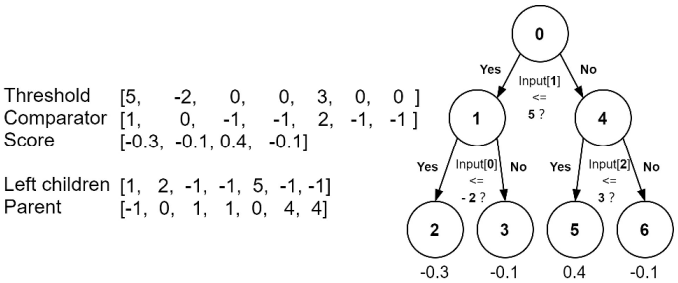


Fig. 3. The binary decision tree structure and its representation in the memory. The index in the memory buffer is the node index in the DT. "-1" indicates the node is a leaf node in the comparators and left children buffers.

The decision-making process of the DT is shown in Fig. 4. The example input is [1, 2, 3]. Every non-leaf node compares its threshold with the corresponding input feature, makes the decision on its leaves to create a decision path, and stores the result in the decision buffer. Then the decision tree will activate its nodes on the correct decision path from the root to leaves. Node 0 is inherently activated. The other nodes are activated when its parent node is activated and the node itself falls on the decision path. Eventually, one of the leaf nodes will be activated, and its score will contribute to the GBDT final result. In this learning module, each DT's score in the ensemble is accumulated to become the final result.

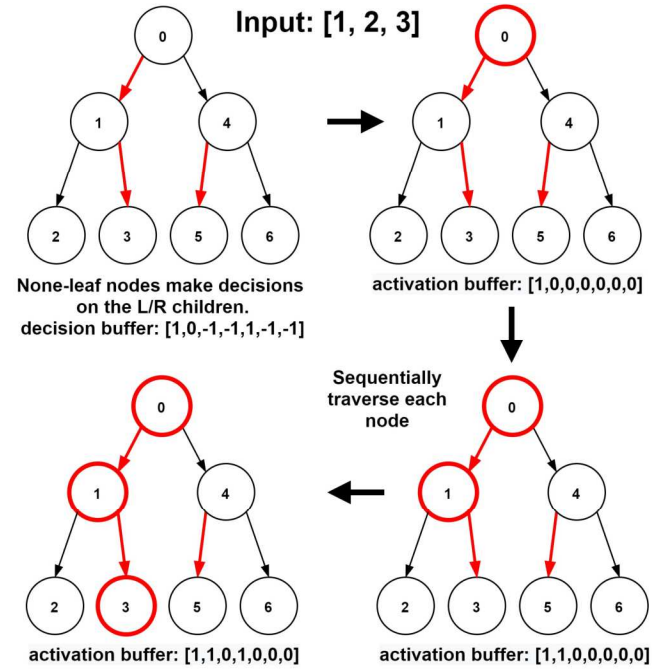
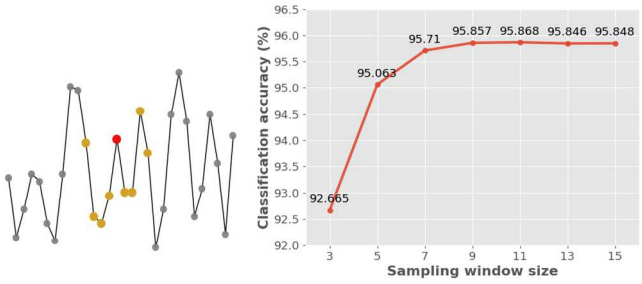


Fig. 4. The decision-making process of the GBDT with input features [1, 2, 3]. The DT first makes the decision on every none-leaf node, then activates the node on the correct decision path in each layer from the root to leaves.

The input features for the BDTI are readback bits processed by anti-aliasing filter (AAF) and digitized by 6-bit ADC. The raw data suffer correlations from the previous signals due to the bandwidth limitation. To detect the target bit's polarity, a sampling window with n prior bits, one target (central) bit, and n subsequent bits are fed into the BDTI as the input features, as shown in Fig. 5(a). The classifier learns the correlations between the $2n + 1$ bits in the sampling window and classifies the central bit's polarity. Given that the worst-case signal-to-noise ratio

(SNR) of the HAMR is 12dB, the BDTI with an ensemble of 200 depth-5 decision trees as weak estimators was tested by using different sampling windows to determine the optimal size for the sampling window. 12 million such samples are partitioned into 10 million training data and 2 million test data. As shown in Fig. 5(b), nine samples are sufficient to represent the sampling windows as a larger window size with increased hardware resources does not give boosts on the performance.



(a) (b)

Fig. 5. (a) An illustration using sampling window (n = 4) to classify the target bit from the noisy bits and (b) the result of testing the target bit classification accuracy using different sampling windows.

Fig. 6 shows the classification accuracy with different numbers of estimators and tree depths. When the tree depth is larger than 4, a GBDT ensemble of m trees with depth h has a higher accuracy comparing with a GBDT of $\frac{m}{2}$ trees with depth of $h+1$. The number of nodes in these two designs is $m*(2^{h+1}-1)$ and $\frac{m}{2}*(2^{h+2}-1)$ correspondingly, which is similar in the total amount of parameters. Therefore, less tree depth and more weak estimators are preferred for better memory efficiency. A GBDT ensemble of 180 trees with depth 5 achieves log bit-error rate (BER) $=-1.38$, which overperforms the conventional method [6] for raw data processing. This model is further tested with various HAMR conditions and the results in Fig. 7 show that the proposed module achieves better classification accuracy with higher SNR data. The BDTI is flexible for extending the number of weak estimators to improve the performance. In this design the baseline-BDTI with 180 trees of depth 5 is implemented in FPGA to achieve real-time classification with low power consumption and low latency.

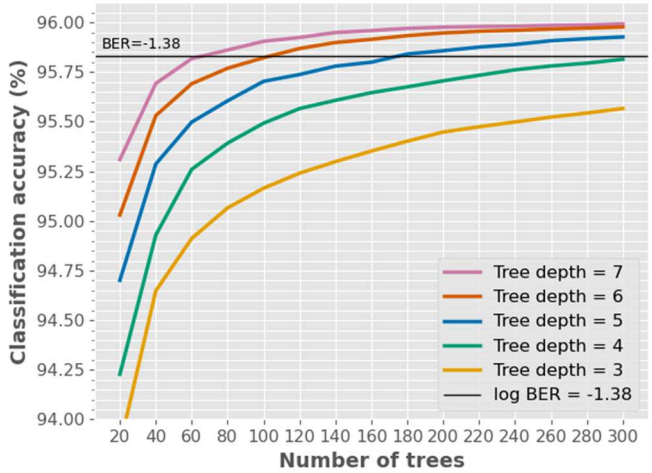


Fig. 6. The classification accuracy for GBDTs with different tree depth and number of trees.

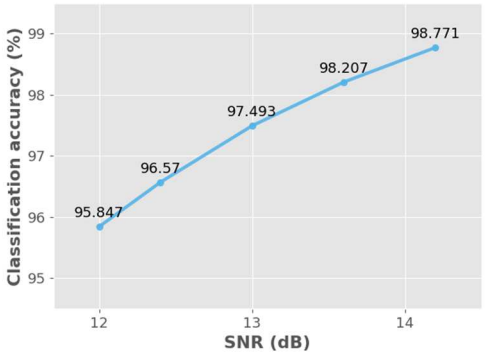


Fig. 7. The BDTI's classification accuracy using data with different signal-to-noise ratios.

IV. Hardware Implementation on FPGA

The proposed BDTI is implemented through Vivado HLS with the GBDT structure and memory organization aforementioned in section III. Fig 8 illustrates the BDTI architecture and the datapath in the FPGA. Note that in the nodes, the comparator only works for non-leaf nodes and the score only works for leaf nodes.

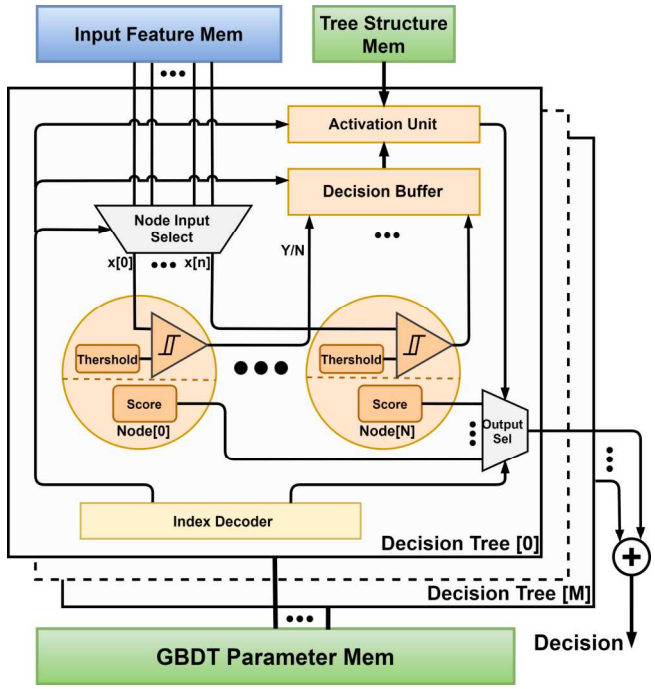


Fig. 8. Hardware architecture of the boosted decision tree interface.

In order to optimize the efficient memory usage, the BDTI parameters are quantized with proper resolutions. The scores in each tree are represented with 16-bit fixed-point numbers. While the classification accuracy with fixed-point representations is comparable to floating-point representations, less hardware resources are required to perform fixed-point operations. The parameters related to the tree node index are using 6-bit integers to represent 64 different node indexes. The 6-bit input data from the ADC have 64 different levels of magnitude(e.g., [1,3,5,…127]), and the thresholds in the decision need to have 65 numbers ([0,2,4,…128]) to compare with the 6-bit input.

Therefore, the input data and the threshold are quantized into 8-bit integers to represent 129 different magnitudes.

A decision tree with depth 5 takes one clock cycle to compare the nodes' thresholds with the input features and five clock cycles to traverse each layer. In this design the decision tree is pipelined by updating the tree parameters to generate the score in every clock cycle. The parallel degree of the BDTI is determined by the number of concurrent trees running in parallel. For M trees running in parallel, the interval of the BDTI inference is $\frac{180}{M} + 5$ clock cycles. In order to determine the parallel degree and optimize the energy and resource efficiency for the design, the equation

$$\frac{\text{Speed}(samples/Sec)}{\text{Power(mW)} * \text{Average hardware resource usage(\%)}}$$

is used to assess the energy-resource efficiency index with different degrees of parallelism of the design. Fig. 9 shows the percentages of the on-chip resource usages, power, speed up, and energy-resource efficiency index versus the different degrees of parallelism in the BDTI. The BDTI runs at 100MHz, and the results are collected from the Vivado synthesis report.

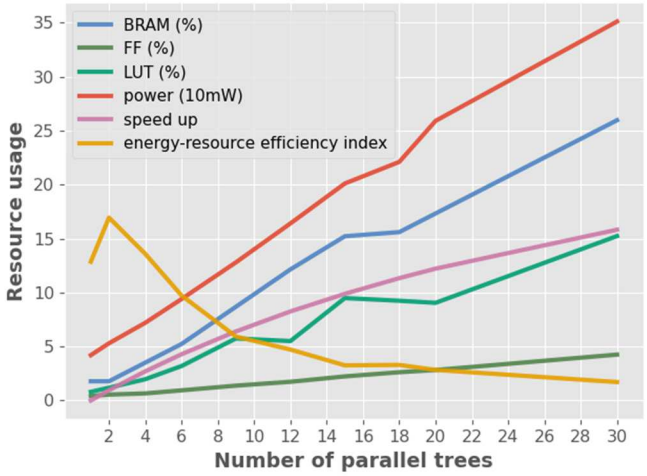


Fig. 9. The on-chip resource usages and performance metrics versus different degrees of parallelism in the boosted decision trees.

As shown in Fig. 9, the BDTI with two trees running in parallel has the best energy-resource efficiency evaluated by using the energy-resource efficiency index. This optimal model was packaged with the AXI4lite-slave protocol in Fig. 10 and deployed on Xilinx MPSoC ZCU102 to evaluate the performance of the proposal readout channel scheme. Fig. 11 shows the setup for the device evaluation. The programmable logic (PL) on ZCU102 is programmed with the BDTI inference intellectual property (IP). A computer communicates with the SoC through UART and transmits the input features to the processing system (PS) to generate the PL output. Then the PS sends the output data back to the computer and the prediction result is displayed on the graphical user interface. The light-weight BDTI model on the FPGA achieves log BER < -1.38 for raw data processing with a flexible architecture, advancing the prior art [6] that utilizes compute-intensive neural networks and does not have a hardware implementation. The module consumes a dynamic power of 53 mW to classify the readback signals with the component SNR of 12dB. The performance is summarized in Table 1.

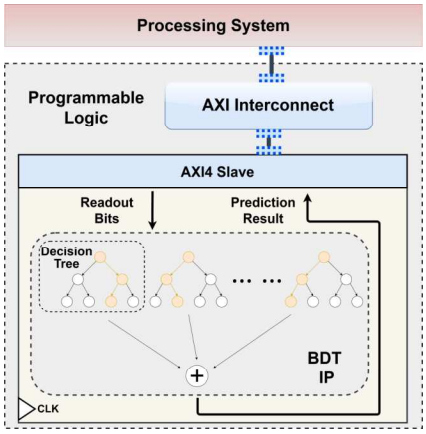


Fig. 10. The system architecture on Xilinx ZCU102.

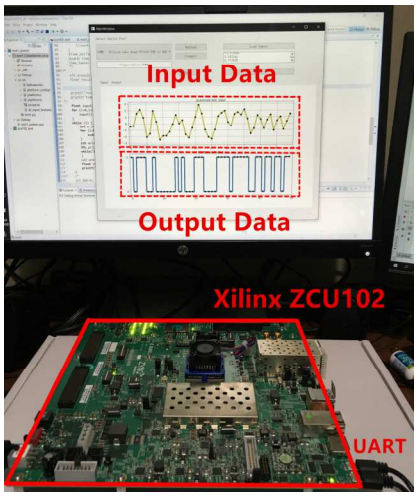


Fig. 11. The hardware setup for real-time data detection with measured classification results versus the raw input data shown on the screen.

TABLE I. PERFORMANCE SUMMARY

Classifier	BDT with Gradient Boosting
Platform	Xilinx ZCU102
Frequency	100 MHz
LUT	3254
FF	3008
BRAM	33
DSP	0
Dynamic Power	53 mW
Classification Accuracy	> 95.8%

V. CONCLUSIONS

Machine-learning-based data symbol detection is presented to overcome noise-and-ISI influence in readout channels to achieve better energy efficiency for multiple actuator applications. The adaptive BDTI modules can learn the high correlation between jitter noise and data patterns to make accurate detections with higher than 95.8% accuracy while consuming only 53mW on the FPGA.

REFERENCES

[1] M. Kryder, E. Gage, and T. McDaniel, "Heat assisted magnetic recording," Proc. IEEE, vol. 96, no. 11, pp. 1810–1835, Nov. 2008.

[2] A. Wu et al., "HAMR areal density demonstration of 1+ Tbpsi on spinstand," IEEE Trans. Magn., vol. 49, no. 2, pp. 779–782, Feb. 2013.

[3] G. Ju et al., "High density heat-assisted magnetic recording media and advanced characterization—Progress and challenges," IEEE Trans. Magn., vol. 51, no. 11, Nov. 2015, Art. no. 3201709, doi: 10.1109/TMAG.2015.2439690.

[4] X. Wang, K. Gao, H. Zhou, A. Itagi, M. Seigler, and E. Gage, "HAMR recording limitations and extendibility," IEEE Trans. Magn., vol. 49, no. 2, pp. 686–692, Feb. 2013.

[5] A. Kavcic and J. M. F. Moura, "The Viterbi algorithm and Markov noise memory," *IEEE Trans. Inf. Theory*, vol. 46, no. 1, pp. 291–301, Jan. 2000.

[6] Y. Qin and J. Zhu, "Deep Neural Network: Data Detection Channel for Hard Disk Drives by Learning," *IEEE Transactions on Magnetics*, vol. 56, no. 2, pp. 1-8, Feb. 2020.

[7] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189-1232, 2001.