Convolutional Neural Network Based Symbol Detector for Two-Dimensional Magnetic Recording

Jinlu Shen[1,2], Benjamin J. Belzer[1], Krishnamoorthy Sivakumar[1], Kheong Sann Chan[3], and Ashish James[4]

[1]School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164-2752 USA
[2]Qualcomm Inc., Santa Clara, CA 95051 USA
[3]Nanjing Institute of Technology, Nanjing 211167, China
[4]Institute for Infocomm Research, A*STAR, Singapore 138632

Data detection in magnetic recording (MR) channels can be viewed as an image processing problem, proceeding from the 2-D image of readback bits, to higher level abstractions of features using convolutional layers that finally allow classification of individual bits. In this work, convolutional neural networks (ConvNets) are employed in place of the typical partial response equalizer and maximum-likelihood detector with noise prediction to directly process the un-equalized readback signals and output soft estimates. Several variations of ConvNets are compared in terms of network complexity and performance. The best performing ConvNet detector with two convolutional layers provides a data storage density of up to 3.7489 Terabits/in^2 on a low track pitch two-dimensional MR channel simulated with a grain-flipping-probability (GFP) model. An alternate ConvNet architecture reduces the network complexity by about 74%, yet results in only a 2.09% decrease in density compared to the best performing detector.

Index Terms— **Convolutional neural network (ConvNet), grain-flipping-probability (GFP) model, machine learning, symbol detection, two-dimensional magnetic recording (TDMR).**

I. Introduction

CONVENTIONAL detection systems in hard disk drives (HDDs) typically include a partial response (PR) equalizer that pre-processes the readback signals and shapes the output to a controlled target response, followed by a maximum-likelihood (ML) sequence detector, such as the Viterbi algorithm [1], or a maximum *a posteriori* (MAP) detector, such as the Bahl–Cocke–Jelinek–Raviv (BCJR) detector [2], which outputs log-likelihood ratios (LLRs) to be passed to a channel decoder. Pattern-dependent noise prediction (PDNP) [3], [4] is usually incorporated into the metric computation of the trellis in the ML/MAP detector to combat media noise intrinsic to the magnetic recording (MR) channel.

Two-dimensional magnetic recording (TDMR) is a promising technology for increasing areal density, where two or more readers are placed on top of one or more closely packed tracks so that the 2-D signal processing can be done to equalize ITI, as opposed to one-dimensional magnetic recording (1DMR) where only one track is processed at a time. For TDMR channels, such conventional Viterbi-PDNP detectors would suffer from impractically large trellis state cardinality when performing multi-track detection, and they may no longer be capable of handling the increased nonlinearities in high-density recording channels.

Research on machine learning applications in MR channels dates back to 1997 [5], in which equalizers based on multilayer perceptron were studied. Recent works have started to investigate more advanced machine learning techniques for MR channels. In [6], a fully connected deep neural network (DNN) detector with no prior PR equalization is employed in a simulated 1DMR channel with transition jitter noise and additive white Gaussian noise (AWGN). In [7], a DNN-based media noise predictor for 1DMR channel is proposed that accepts the BCJR's LLR estimates as input and outputs media noise estimates to be passed back to the BCJR. Both fully connected DNNs and convolutional neural networks (ConvNets) are studied in [7]. In [8], a nonlinear PR equalizer based on fully connected neural network structure and adapted with cross-entropy criterion between the true probability of the bit and detector's estimate of it is proposed.

In [9], we investigate a DNN-based detector with prior PR equalization for TDMR channels simulated on a realistic channel model called grain-flipping-probability (GFP) model. Three types of DNN architectures are studied, including fully connected DNN, ConvNet, and long short-term memory (LSTM) network, a type of recurrent neural network. Simulation results show that all three DNN architectures yield significant detector bit error rate (BER) reductions over a 2-D PDNP system [10] and a local area influence probabilistic (LAIP)-BCJR system [11]–[13]. It is also shown in [9] that ConvNet achieves the lowest detector BER as well as the highest code rate and informational areal density.

In this work, we further study a stand-alone ConvNet detection system without prior PR equalization, which directly processes the un-equalized readback signals and outputs soft estimates. ConvNets are special DNNs that assume the inputs are images and perform convolution instead of applying affine functions in the network's forward pass [2]. This enables far fewer parameters in ConvNets than regular DNNs of

Manuscript received August 3, 2020; revised September 14, 2020; accepted October 31, 2020. Date of publication November 3, 2020; date of current version February 18, 2021. Corresponding author: J. Shen (e-mail: jinlu.shen@ieee.org).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TMAG.2020.3035705.

Digital Object Identifier 10.1109/TMAG.2020.3035705

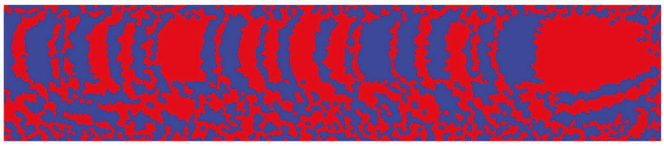

Fig. 1. Capture of the readback signals simulated from the GFP model.

the same depth and therefore allows for deeper networks. The motivation to use ConvNets is the resemblance between the MR data detection problem and the typical image processing problems. In MR channels, the write process converts temporal data into spatial patterns recorded on a magnetic medium, which transforms sequential correlation into spatial ISI/ITI. Data detection in MR channels can thus be viewed as an image processing problem, proceeding from the 2-D image of the shingled bits (see Fig. 1), to higher level abstractions of features using convolutional layers that finally allow classification of individual bits.

To our knowledge, this article is one of the first to employ a stand-alone ConvNet detector and to combine it with a channel decoder to obtain achieved code rate/areal density results.

II. System Model

The ConvNet detection system assumes a discrete channel model for the kth readback signal r_k

$$r_k = (\mathbf{h} * \mathbf{u})_k + n_{e,k} \tag{1}$$

where $\mathbf{h}$ is the 2-D channel response, $\mathbf{u}$ is the 2-D block of coded bits, $*$ indicates discrete 2-D convolution, and $n_{e,k}$ is reader electronics, modeled as AWGN. The channel response $\mathbf{h}$ is implicitly time-varying and pattern-dependent because the channel is inherently nonlinear, thus giving rise to pattern-dependent noise.

In our experiments, the system is trained and tested using data from a GFP model, a realistic channel model that closely replicates output from micro-magnetic simulations but can be generated several orders of magnitude faster [14]. Several existing works, including [15]–[18], provide validation of the GFP model. The simulated media has a grain density of 11.4 Teragrains/in^2. The GFP data include five tracks of $N_b = 41{,}206$ coded bits (tracks 0–4); only the three central tracks (tracks 1–3) have available readback signals. Fig. 1 shows a capture of the GFP readback signals. The blue and red stripes represent -1 and $+1$ coded bits, respectively. They are curved and overlapping each other because of the shingled writing process. Track 0 was written first and track 4 was written last. The combination of white rectangles on top of the stripes indicates a convolutional filter applied on the data, and the white arrow indicates the direction that this filter moves (further explanation in Section III). The ConvNet system is tested on two GFP data sets, each containing 100 simulated blocks. The two data sets have the same bit length (BL) of 11 nm, but different track pitches (TPs): TP 15 nm [equivalently 2.916 grains per coded bit (GPB)] and TP 18 nm (equivalently 3.491 GPB).

III. ConvNet Detector

Fig. 2 shows the block diagram for the proposed ConvNet detection system. Three identically structured ConvNets

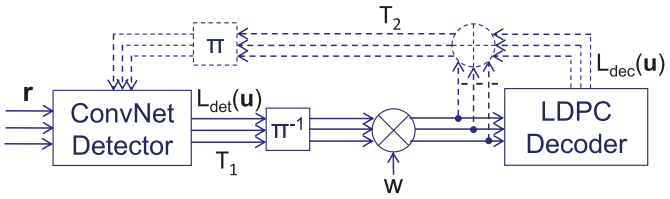

Fig. 2. Block diagram for the ConvNet detection system.

estimate tracks 1–3 in parallel, each in a downtrack sliding window. The ConvNet detector processes a 2-D patch of readback signals $\mathbf{r}$ from three tracks and outputs a probability estimate for the center bit of the patch. These soft estimates are converted into LLRs $L_{\text{det}}(\mathbf{u})$ before being magnitude limited to T_1, de-interleaved (π^{-1}), weighted by W, and passed to a low-density parity check (LDPC) decoder. The dotted lines and box in Fig. 2 indicate future work, i.e., iterative detection between the ConvNet detector and the LDPC decoder, where extrinsic information components of the decoder output LLRs $L_{\text{dec}}(\mathbf{u})$ get magnitude limited to T_2, interleaved (π) and passed back to the ConvNet detector.

The ConvNet detector has a general architecture of [INPUT - CONV - RELU - CONV - RELU - FC - softmax] in forward pass. The INPUT layer denotes the input to the network. It is a matrix representation of an image-like object: $[3, w_{\text{in}}]$, where three indicates three tracks and w_{in} denotes the width of the input. The GFP data are shaped into many examples of dimension $[3, w_{\text{in}}]$ prior to ConvNet processing. The bit to be classified per example is the one at the center of the 2-D image, i.e., the one with coordinate $(1, ((w_{\text{in}} - 1)/2))$ if the point at the top-left corner has coordinate $(0, 0)$; here, w_{in} is always an odd number. For detection of tracks 1 and 3, binary bits on boundary track (tracks 0 and 4) are used to form the image object. Although a typical image object in ConvNet usually has a third dimension representing the number of color channels, i.e., whether it is a red/green/blue (RGB) color channel or a black/white (BW) channel, there is no such physical implication in the MR channel, and we simply set the third input dimension to 1. Note that for iterative decoding investigated in [9], the image input in the second iteration has a third dimension of 2, considering both the readback values and the channel decoder's estimate of the bits in the first iteration. CONV layer #k computes dot products between c_k different trained filters of size $[3, w_{c_k}]$ (black rectangles in Fig. 3) and the same-sized patches in its input (red solid rectangles in Fig. 3), with the filters moving in a sliding window fashion in both horizontal (downtrack) and vertical (crosstrack) direction of the layer input. Assuming that the CONV layer input dimension is $[h, w]$, then the dimension of each filter's output at this CONV layer is also $[h, w]$, due to zero padding (outermost red dotted rectangles in Fig. 3) that preserves its input dimension. Therefore, the total output dimension of CONV layer #k output is $[h, w, c_k]$.

The RELU layer applies the rectified linear unit (ReLU) activation function $f(z) = \max(0, z)$ element-wise on its input z. Between the CONV and RELU layers, a batch normalization layer is added to normalize the mean and variance per mini batch of CONV layer output, thereby accelerating the

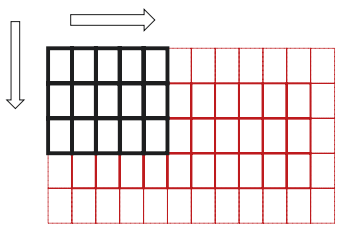

Fig. 3. Convolutional filters (thick black rectangles at the top-left corner) of CONV layer moving across layer input (solid red rectangles) with zero padding (outermost thin dotted red rectangles).

training. This introduces two network learnables β and γ as defined in [19]. At the end of training, the mean and variance over all training examples are calculated and stored; these values are loaded during inference. Experiments show that two stacks of CONV-RELU layers are sufficient to yield a low BER or 10^{-5} or lower. FC stands for fully connected layer, in which each node x_k at the current layer is connected to all nodes z_j in the previous layer's output, and an affine function $x_k = \sum_j w_{jk} \cdot z_j + b_k$ is applied to them, where w_{jk} are trainable connection weights. A probability estimate p_{ik} of the center bit in each example is then formed using the softmax function $p_{ik} = \exp(x_k^{(i)}) / \sum_{k=1}^{2} \exp(x_k^{(i)})$, where i represents the ith example, k denotes which class it belongs to, and p_{ik} indicates the probability that the ith example belongs to class k. Since coded bit detection is a binary classification problem, k can assume two values; $k = 1$ represents a -1 bit and $k = 2$ a $+1$ bit. During the training phase, the ConvNet is optimized in backward pass using gradient descent with cross-entropy loss $J = \sum_{i=1}^{N_{\text{mb}}} \sum_{k=1}^{2} 1(\hat{u}(i) = k) \cdot \log(p_{ik})$, where $1(\cdot)$ denotes the indicator function, which evaluates to 1 when its input is true, and $\hat{u}(i)$ denotes the ConvNet detector's estimation of the coded bit $u(i)$ (target bit of the ith example). The network is updated every mini batch of N_{mb} training examples, which accelerates the training compared to processing all training examples in one single batch. Our choice of optimizer is the adaptive moment estimation (Adam) optimizer [20], which is a variant of classical stochastic gradient descent that utilizes an adaptive learning rate to improve performance. At the end of training, the ConvNet stores all the learnables that have been optimized. During the inference phase, the ConvNet loads all the learnables and computes in forward pass its probability estimates of the center bits in testing examples.

IV. Simulation Results

The proposed ConvNet symbol detector is tested on the two GFP model data sets with different TPs described in Section II. The raw BERs of readback signals r_k are roughly 20.44% for TP 15 nm data set and 16.41% for TP 18 nm data set. For each GFP data set, of the 100 blocks available, 80 blocks are used for training and the remaining 20 blocks for inference. The total number of training examples is thus $80 \cdot (N_b - w_{\text{in}} + 1)$. In the Adam optimizer, default values for hyperparameters β_1, β_2, and ϵ as suggested in [20] are used: $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-8}$. The learning rate is chosen to be $\mu = 10^{-3}$. For the batch normalization layer, default value for hyperparameter ϵ_{bn} is used: $\epsilon_{bn} = 10^{-5}$. Network is updated every mini batch of $N_{\text{mb}} = 10{,}000$ training examples, resulting in roughly 330 iterations in one training epoch (during which one pass of going through all training examples is completed), given the values of w_{in} in our experiments. The network is trained for 30 epochs, which we observe to be sufficient.

TABLE I

ConvNet BER and Storage Complexity, TP 15 nm, Track 2

Network Structure	BER train-test	# learnables
[3,11], [3,11]×128, [3,9]×64	0.0060-0.0118	230,210
[3,15], [3,11]×128, [3,9]×64	0.0021-0.0053	231,746
[3,17], [3,11]×128, [3,9]×64	0.0016-0.0040	232,514
[3,21], [3,11]×128, [3,9]×64	**0.0011-0.0029**	**234,050**
[3,21], [3,11]×64, [3,9]×32	**0.0035-0.0068**	**61,730**
[3,21], [3,9]×128, [3,7]×64	0.0017-0.0041	184,130
[3,21], [3,11]×32, [3,9]×16	0.0186-0.0213	17,042

TABLE II

Overall ConvNet Density Performance

TP	ConvNet Structure	Average Code Rate	Average User Areal Density (Tb/in^2)
15	#1: [3,21], [3,11]×128, [3,9]×64	0.9567	**3.7489** (2.6278)
15	#2: [3,21], [3,11]×64, [3,9]×32	0.9367	**3.6706** (2.5728)
18	#1: [3,21], [3,11]×128, [3,9]×64	0.9783	3.1949 (2.3569)
18	#2: [3,21], [3,11]×64, [3,9]×32	0.9700	3.1677 (2.3368)
18	ConvNet with PR equalizer [9]	0.9467	3.0915 (2.2806)

Table I summarizes the BER performance and storage complexity of ConvNets trained with different values of w_{in}, w_{c_k}, and c_k, for track 2 of TP 15 nm data set. The first column, network structure, lists from left to right the size of INPUT layer $[3, w_{\text{in}}]$, the dimension of CONV layer #1 $[3, w_{c_1}] \times c_1$, and the dimension of CONV layer #2 $[3, w_{c_2}] \times c_2$. The second column indicates training and inference BER at the ConvNet output. The third column shows the total number of learnables in the network. Comparing the first four rows, input width $w_{\text{in}} = 21$ yields the lowest BER. Comparing the last four rows, we conclude the number of filters c_k at each CONV layer has the most influence on both storage complexity and performance. Specifically, rows 4 and 5 show that reducing both c_1 and c_2 by half lowers storage complexity by roughly a factor of 3.8 but only suffers a factor of 3 increase in BER. The lowest BER, 0.0029, is achieved from a ConvNet of 234 050 learnables shown in row 4. The ConvNet structure in row 4—[3,21], [3,11] × 128, [3,9] × 64 is denoted as ConvNet #1 and that in row 5—[3,21], [3,11] × 64, [3,9] × 32 is denoted as ConvNet #2. ConvNets #1 and #2 are used for further analysis in this article.

The ConvNet detector output LLRs are passed to an irregular repeat accumulate (IRA) decoder. IRA codes are a special type of LDPC codes that have linear encoding time. Because the GFP data set is random rather than encoded, coset decoding is employed. Table II shows the average achieved code rate $\bar{R}$ and user areal density (UAD) results over tracks 1–3 from ConvNet #1 and ConvNet #2, for both TP 15 nm and TP 18 nm data sets. The achieved code rate R per track is the highest

TABLE III
DETECTOR AND DECODER BER, TP 18 nm

Detector	Average Detector BER	Decoder BER
2D-PDNP	6.4856e-2	0(6.8377e-6)
ConvNet with PR equalizer [9]	6.9667e-3	0(1.6153e-6)
ConvNet #1	2.1333e-3	0(1.6153e-6)
ConvNet #2	3.5333e-3	0(1.6153e-6)

TABLE IV
COMPUTATIONAL COMPLEXITY COMPARISON

Detector	mul/div	add/sub	exp/log
2D-PDNP	4,481	4,216	257
ConvNet with PR equalizer [9]	1,687,850	1,687,850	3
ConvNet #1	5,045,826	5,045,826	3
ConvNet #2	1,311,522	1,311,522	3

code rate after puncturing data on the track that achieves a final decoded BER of 10^{-5} in the Monte Carlo simulations. The puncturing scheme employed in the system simulates practical puncturing in an HDD and is described in detail in [12]. The UAD per track is calculated as UAD $= R/\text{GPB}\cdot\text{Grain-density}$, where GPB and Grain-density are specified in Section II. The unit of UAD is Terabits/in^2 (Tb/in^2). The UAD values in parentheses are scaled density values after a 6.4 nm squeeze margin, i.e., $\text{UAD}_{\text{sm}} = \text{UAD} \cdot (\text{TP}/(\text{TP} + 6.4))$. The highest density before squeeze margin, 3.7489 Tb/in^2, is achieved on TP 15 nm data set, as shown in row 1. To the best of our knowledge, this is one of the highest densities ever reported on GFP model data with a grain density of 11.4 Teragrains/in^2. It is 21.27% higher than the highest density of 3.0915 Tb/in^2 achieved by the ConvNet designed in [9] in row 5 (the number reported in [9] is 3.081; a more optimal IRA code as used in this article updates it to 3.0915). Furthermore, the first two rows of Table II show that for TP 15 nm data set, a 2.0905% degradation in UAD performance can be exchanged for a factor of 3.8 decrease (74% decrease) in complexity, whereas rows 3 and 4 show that for TP 18 nm data set, only 0.8484% performance degradation results from the same amount of decrease in complexity. Such complexity–performance trade-off is helpful for hardware implementation.

Table III shows the average detector BER and decoder BER at highest achieved code rate in one shot for the proposed ConvNets #1 and #2, a 2-D PDNP detector and a different ConvNet with prior PR equalizer. The 2-D PDNP detects two tracks simultaneously, using the following 2-D autoregressive model to predict media noise [10]:

$$\mathbf{n}_k = \sum_{i=0}^{N_p} \mathbf{P}_i(\mathbf{A}_k)\mathbf{n}_{k-i} + \Lambda(\mathbf{A}_k)\mathbf{w}_k \tag{2}$$

where $\mathbf{n}_k$ denotes the two-element vector of predicted noise samples from both tracks at downtrack position k, N_p denotes the order of predictor, $\mathbf{A}_k$ denotes the pattern matrix, $\mathbf{P}_i(\mathbf{A}_k)$ denotes the coefficient matrix, $\Lambda(\mathbf{A}_k)$ denotes the standard deviation matrix, and $\mathbf{w}_k \sim \mathcal{N}(0, \mathbf{I})$ is a vector of AWGN. Details of our implementation of 2-D PDNP are described in [13]. Details for the ConvNet with prior PR equalizer are presented in [9]. The number in parenthesis a conservative BER upper bound estimates with a 95% confidence level when the error count is 0, computed as $3/N_{\text{tcb}}$, where N_{tcb} is the total number of transmitted information bits [21]. The BERs of the proposed ConvNets #1 and #2 are 69.67% and 49.28% less than that of the ConvNet in [9] with prior PR equalizer, as well as factors of 1/30 and 1/18 that of 2-D PDNP.

Because the GFP data set contains no AWGN to account for read-head electronics, we manually add AWGN at an SNR of 20 dB using the following formula:

$$\text{SNR} = 10 \cdot \log_{10} \frac{E\left[r_k^2\right]}{2 \cdot R \cdot \sigma_n^2} \tag{3}$$

where σ_n^2 is the variance of the AWGN and R is the code rate. No performance degradation is found in the average code rate and UAD numbers reported in row 1 of Table II. For rows 2–4, there are 0.7117%, 0.3407%, and 1.3746% reductions, respectively. Future work could consider a more realistic boundary condition, i.e., assuming certain prior BER for the input bits on the two boundary tracks (tracks 0 and 4).

Table IV compares the computational complexity (per bit) of the proposed ConvNets with 2-D PDNP and the ConvNet in [9] with prior PR equalizer. We point out that the numbers of computations for the ConvNet with prior PR equalizer reported in [9, Table IV] should be multiplied by a factor of 15; they are reported correctly in row 2 of Table IV. We also consider a more efficient implementation of trellis-based 2-D PDNP, leading to different numbers of computations reported in row 1 of Table IV than in [9, Table IV]. The relatively short detection time of the ConvNet with priori PR equalizer reported in [9] is likely due to MATLAB's vector implementation (which accelerates equivalent codes implemented with loops). Table IV shows that all the ConvNets require several orders of magnitude more computations than conventional 2-D PDNP—a price paid for superior performance. The computational complexity required by ConvNet #1 is roughly 3× that of the ConvNet in [9] with PR equalizer, in exchange for 69.67% detector BER decrease (see Table III) and equivalently 3.34% areal density increase (see Table II). On the other hand, ConvNet #2 requires 24.13% less computational complexity than the ConvNet in [9] with PR equalizer, yet still gives 49.28% detector BER decrease and 2.46% areal density increase.

V. CONCLUSION

We design a ConvNet-based symbol detector for TDMR that involves only two convolutional layers and achieves UAD as high as 3.7489 Tb/in^2 on densely recorded GFP model data. The proposed system suffers from as low as 0.8484% performance degradation when another ConvNet with 74% lower complexity than the best performing architecture is used.

ACKNOWLEDGMENT

This work was supported by the United States National Science Foundation (NSF) under Grant CCF-1817083.

REFERENCES

[1] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vol. IT-13, no. 2, pp. 260–269, Apr. 1967.

[2] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (Corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-20, no. 2, pp. 284–287, Mar. 1974.

[3] 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.

[4] J. Moon and J. Park, "Pattern-dependent noise prediction in signal-dependent noise," *IEEE J. Sel. Areas Commun.*, vol. 19, no. 4, pp. 730–743, Apr. 2001.

[5] S. K. Nair and J. Moon, "Data storage channel equalization using neural networks," *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 1037–1048, Sep. 1997.

[6] Y. Qin and J.-G. Zhu, "Deep neural network: Data detection channel for hard disk drives by learning," *IEEE Trans. Magn.*, vol. 56, no. 2, pp. 1–8, Feb. 2020, Art. no. 6701108.

[7] A. Sayyafan, B. J. Belzer, K. Sivakumar, J. Shen, K. S. Chan, and A. James, "Deep neural network based media noise predictors for use in high-density magnetic recording turbo-detectors," *IEEE Trans. Magn.*, vol. 55, no. 12, pp. 1–6, Dec. 2019, Art. no. 6701406.

[8] J. Shen and N. Nangare, "Nonlinear equalization for TDMR channels using neural networks," in *Proc. 54th Annu. Conf. Inf. Sci. Syst. (CISS)*, Mar. 2020, pp. 1–6.

[9] J. Shen, A. Aboutaleb, K. Sivakumar, B. J. Belzer, K. S. Chan, and A. James, "Deep neural network a posteriori probability detector for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 56, no. 6, pp. 1–12, Jun. 2020.

[10] S. Shi and J. R. Barry, "Multitrack detection with 2D pattern-dependent noise prediction," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[11] X. Sun, K. Sivakumar, B. J. Belzer, and R. Wood, "High density turbo TDMR detection with local area influence probabilistic model," *IEEE Trans. Magn.*, vol. 53, no. 2, pp. 1–8, Feb. 2017, Art. no. 9400208.

[12] X. Sun *et al.*, "ISI/ITI turbo equalizer for TDMR using trained local area influence probabilistic model," *IEEE Trans. Magn.*, vol. 55, no. 4, pp. 1–15, Apr. 2019.

[13] J. Shen, X. Sun, K. Sivakumar, B. J. Belzer, K. S. Chan, and A. James, "TDMR detection system with local area influence probabilistic *a priori* detector," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2019, pp. 1–7.

[14] K. S. Chan *et al.*, "Channel models and detectors for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 46, no. 3, pp. 804–811, Mar. 2010.

[15] K. S. Chan, E. M. Rachid, K. Eason, R. Radhakrishnan, and K. K. Teo, "Comparison of one- and two-dimensional detectors on simulated and spin-stand readback waveforms," *J. Magn. Magn. Mater.*, vol. 324, no. 3, pp. 336–343, Feb. 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304885310009170

[16] K. K. Teo, M. R. Elidrissi, K. S. Chan, and Y. Kanai, "Analysis and design of shingled magnetic recording systems," *J. Appl. Phys.*, vol. 111, no. 7, 2012, Art. no. 07B716. [Online]. Available: http://aip.scitation.org/doi/pdf/10.1063/1.3679383

[17] K. S. Chan *et al.*, "Comparison of signals from micromagnetic simulations, GFP model, and an HDD readback," *IEEE Trans. Magn.*, vol. 51, no. 11, pp. 1–4, Nov. 2015.

[18] M. R. Elidrissi *et al.*, "Modeling of 2-D magnetic recording and a comparison of data detection schemes," *IEEE Trans. Magn.*, vol. 47, no. 10, pp. 3685–3690, Oct. 2011.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: http://arxiv.org/abs/1502.03167

[20] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: http://arxiv.org/abs/1412.6980

[21] F. Scholz. (2008). *Confidence Bounds and Intervals for Parameters Relating to the Binomial, Negative Binomial, Poisson and Hypergeometric Distributions With Applications to Rare Events*. ConfidenceBounds.pdf. [Online]. Available: http://faculty.washington.edu/fscholz/DATAFILES498B2008/