Turbo-Detection for Multilayer Magnetic Recording Using Deep Neural Network-Based Equalizer and Media Noise Predictor

Amirhossein Sayyafan[1], Ahmed Aboutaleb[1], Benjamin J. Belzer[1], Krishnamoorthy Sivakumar[1],
Simon Greaves[2], and Kheong Sann Chan[3]

[1]School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164 USA
[2]Research Institute of Electrical Communication (RIEC), Tohoku University, Sendai 980-8577, Japan
[3]Department of Electrical and Electronic Engineering, University of Nottingham Malaysia, Semenyih 43500, Malaysia

This article considers deep neural network (DNN)-based turbo-detection for multilayer magnetic recording (MLMR), an emerging hard disk drive (HDD) technology that uses vertically stacked magnetic media layers with readers above the top-most layer. The proposed system uses two layers with two upper layer tracks and one lower layer track. The reader signals are processed by convolutional neural networks (CNNs) to separate the upper and lower layer signals and equalize them to 2-D and 1-D partial response (PR) targets, respectively. The upper and lower layer signals feed 2-D and 1-D Bahl–Cocke–Jelinek–Raviv (BCJR) detectors, respectively. The detectors' soft outputs feed a multilayer CNN-based media noise predictor whose predicted noise outputs are fed back to the BCJR equalizers to reduce their bit error rates (BERs). The BCJR equalizers also interface with low-density parity-check (LDPC) decoders. Additional BER reductions are achieved by sending soft-information from the upper layer BCJR to the lower layer BCJR. Simulations of this turbo-detection system on a two-layer MLMR signal generated by a grain-switching-probabilistic (GSP) media model show density gains of 11.32% over a comparable system with no lower layer and achieve an overall density of 2.6551 terabits per square inch (Tb/in^2).

***Index Terms*— Bahl–Cocke–Jelinek–Raviv (BCJR) detector, convolutional neural network (CNN), CNN equalizer-separator, CNN media noise predictor, deep neural network (DNN), low-density parity-check (LDPC) decoder, multilayer magnetic recording (MLMR), turbo-detector.**

I. Introduction

MULTILAYER magnetic recording (MLMR), proposed in [1]–[5], is an emerging technology to increase the areal information density of hard disk drives (HDDs) using vertically stacked magnetic media layers. The respective layers use different bit sizes and can be independently written at different resonant frequencies using microwave assisted magnetic recording (MAMR). The presence of lower layers gives a net areal density (AD) gain compared with the case where only the upper layer is present.

In MLMR, the readers are above the top-most layer; hence, the readback signal is a superposition of signals from all the layers, resulting in a joint signal separation and equalization problem. This problem can be reduced by track interlacing such that upper layer tracks only partially overlay lower level tracks, as in the two-layer interlaced MLMR considered in [6] and [7], but at the cost of less potential AD gain than the two-layer MLMR considered in [8] and [9] and in the present article, wherein two upper layer tracks completely cover one lower layer track. Thus, while reader signal processing techniques used in two-dimensional magnetic recording (TDMR) [10], such as, e.g., [11]–[19] can potentially be used to increase the per-layer AD in MLMR, such techniques can only be used after separation of the individual MLMR layer signals is performed.

By viewing TDMR signals as 2-D images, TDMR detection can be recast as an image classification problem, making it suitable for solution via deep neural networks (DNNs). DNN architectures such as fully connected deep neural networks (FCDNNs) and especially convolutional neural networks (CNNs) have compiled impressive results in image classification and understanding [20]. A number of recent articles (e.g., [15]–[19], [21], [22]) have applied DNN-based signal processing to TDMR detection, and a few articles (e.g., [7]–[9]) have applied DNNs to detection of two-layer MLMR signals.

The CNN-based equalizer-separator used in the present article was first proposed in [8] for a similar MLMR media stack with two upper layer tracks completely overlaying one lower layer track; however, the upper and lower layer track pitches (TPs) in [8] were 48 and 96 nm, respectively, while the bit lengths (BLs) were 11 and 22 nm, respectively, so that there were four upper layer bits per lower layer bit. The present article has upper and lower layer TPs of 24 and 48 nm, respectively, and BLs of 10 and 20 nm, respectively, allowing significantly higher ADs than those achieved in [8]. The CNN equalizer-separator outputs in [8] were separated into three 1-D signals (two upper layer track signals and one lower layer track signal) and equalized to three 1-D partial response (PR) targets. The per-track PR equalized signals were fed into three separate 1-D soft-output Viterbi algorithm (1-D SOVA) detectors, and the final information bits were obtained by decoding the SOVA detector outputs with two low-density parity-check (LDPC) decoders, one for the two upper layer tracks (which used identical LDPC codes) and one for the

Manuscript received August 4, 2021; accepted October 12, 2021. Date of publication October 22, 2021; date of current version March 18, 2022. Corresponding author: B. J. Belzer (e-mail: belzer@wsu.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TMAG.2021.3122136.

Digital Object Identifier 10.1109/TMAG.2021.3122136

lower layer track. The system in [8] attempted to remove both the interlayer interference (ILI) between the layers and the intertrack interference (ITI) between the two top-level tracks. Also, the system in [8] did not include explicit means to predict or cancel data-dependent media noise.

In the present article, the CNN equalizer-separator equalizes the two upper layer tracks to a 2-D PR target and the lower layer track to a 1-D PR target. The upper and lower layer signals feed into 2-D and 1-D Bahl–Cocke–Jelinek–Raviv (BCJR) [23] detectors, respectively. Thus, the CNN equalizer-separator in the present article focuses only on removing ILI and leaves ITI removal to the 2-D BCJR detector. The BCJR detector outputs feed a multilayer CNN media noise predictor, which in turn outputs media noise predictions for the upper and lower layer tracks. By accepting inputs from both BCJR detectors, the CNN media noise predictor can account for 3-D media noise effects between layers, as well as intra-layer media noise. The media noise predictions are subtracted from the per-branch conditional means in a second pass through both BCJR detectors. Then the BCJR outputs are decoded by LDPC decoders for the upper and lower layer tracks. The bit error rate (BER) of the lower layer BCJR detector is further reduced by passing log-likelihood ratios (LLRs) for the upper layer bits from the upper layer 2-D BCJR to the lower layer 1-D BCJR and using them as soft-decision feedback to remove residual ILI from the lower layer bits. An architecture that combined a 2-D BCJR detector with CNN noise predictor and LDPC decoder was previously used for turbo-detection of three-track TDMR signals in [18], where it achieved an information AD of 3.88 Tb/in^2 on a grain-flipping-probabilistic (GFP) media model (which is based on micro-magnetic simulations) with 15 nm TP and 11 nm BL.

Both the present article and [8] train and test their turbo-equalization systems with MLMR reader waveforms generated by a grain-switching-probabilistic (GSP) model. The GSP model, developed in [1] and [2], is a grain-level model that generates MLMR waveforms using a micromagnetic model to train multi-dimensional conditional probability mass functions that determine the probability that a given grain will switch in response to an applied write field. The GSP models enable relatively fast generation of very large MLMR datasets. In addition to incorporating the superposition of signals from the upper and lower layers (including the ITI) in the readback waveforms, the GSP model uses realistic media noise models and incorporates transition noise during the write process that was not considered in previous models.

The contributions of the present article are as follows.

1) A turbo-detection architecture that combines the CNN-based equalizer-separator of [8] with the 1-D/2-D BCJR detector and CNN media noise predictor of [18] and includes innovations to both these components to enable improved detection of two-layer MLMR signals.
2) The equalizer-separator equalizes the upper layer to a 2-D PR target, and the upper layer signal is processed by a 2-D BCJR detector; hence, the equalizer-separator focuses only on ILI removal and allows the 2-D BCJR to focus on ITI removal for the upper-layer signal.
3) The 3-D CNN media noise predictor accepts BCJR outputs from both the layers, enabling it to account for both inter- and intra-layer media noise and thereby improve its noise prediction for both layers.
4) The upper layer 2-D BCJR passes LLRs to the lower layer 1-D BCJR, which the 1-D BCJR uses as soft-decision feedback to reduce residual ILI from upper layer bits.
5) The simulation results on the GSP-simulated MLMR media with two upper layer tracks at 24 nm TP and 10 nm BL and one lower layer track at 48 nm TP and 20 nm BL show an 11.32% information AD gain over the case where only the upper layer is present and also show a 6.63% density gain over the case where 1-D pattern-dependent noise prediction (PDNP) is used to detect individual upper and lower layer tracks equalized to 1-D PR targets by the CNN equalizer-separator.

II. System Model

The BCJR-LDPC-CNN turbo-detector assumes a channel model for the kth output of the lth CNN equalizer-separator $\hat{\mathbf{s}}_l(k), l \in \{1, 2\}$

$$\hat{\mathbf{s}}_l(k) = (\mathbf{h}_l * \mathbf{u})(k) + n_m(k) + n_e(k) \qquad (1)$$

where $\mathbf{h}_l$ is the PR target, $\mathbf{u}$ are the coded bits on the track, * indicates 1-D/2-D convolution, $n_m(k)$ is the media noise, and $n_e(k)$ is the reader electronics additive white Gaussian noise (AWGN). Unlike PDNP, the media noise term $n_m(k)$ is not modeled as an autoregressive (AR) process; instead, a more general model for $n_m(k)$ is learned by the CNN noise predictor.

We use realistic GSP model data to train and evaluate our system. The GSP waveforms are generated based on micro-magnetic simulations of microwave-assisted magnetic recording (MAMR) [1], [2].

Fig. 1 illustrates a two-layer recording structure. The lower layer is farther apart from the read head compared with the upper layer, resulting in a weaker signal from the lower layer. To compensate for this, the bit area on the lower layer is four times larger than that on the upper layer to allow for sufficient signal power from the lower layer at the reader. Moreover, there is signal attenuation due to the upper layer material between the lower layer and the reader. In general, the ratio of the number of bits on the upper layer to that on the lower layer is a system parameter.

MLMR requires joint signal separation and equalization: the readers lie just above the disk surface and hence receive a superposition of signals from the upper and lower layers, and the received signal now suffers from 3-D-intersymbol interference (ISI) due to per-layer ITI and down-track ISI, plus ILI.

The cross-track view of the MLMR system is represented in Fig. 2. On the upper layer, six tracks at TP 24 nm and BL 10 nm are written. There are three tracks on the lower layer written at TP 48 nm and BL 20 nm. Hence, for each bit on the lower layer, four bits are stored on the upper layer. The bit sequences written on the upper left and right tracks are denoted by $\mathbf{a}_{2,L}$ and $\mathbf{a}_{2,R}$, respectively, which are the tracks of

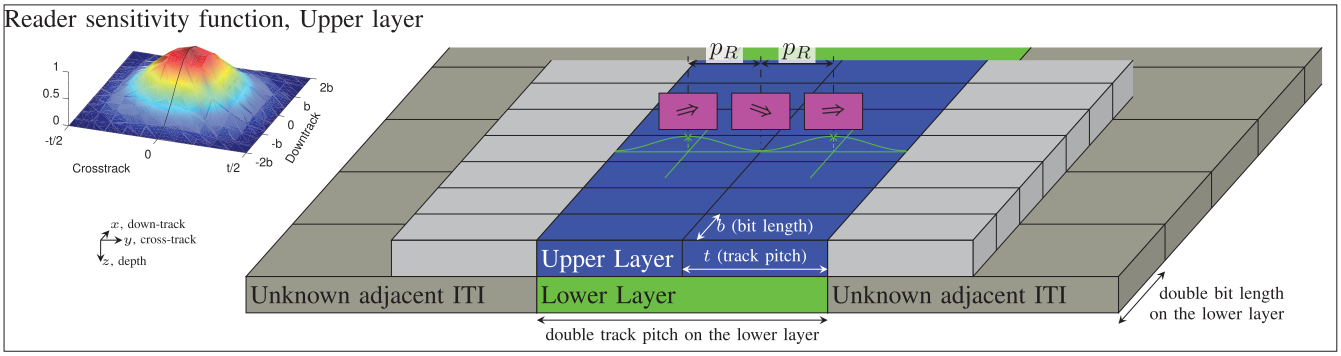

Fig. 1. Two-layer magnetic recording structure. The 2-D reader sensitivity function gives the 2-D response of the read head. Bits on the lower layer are written at quarter density compared with the upper layer, as indicated by the double BL and TP. Although omitted here, the two-layer recording structure includes a non-magnetic material in-between layers and a soft magnetic underlayer underneath the lower layer [5], [9].

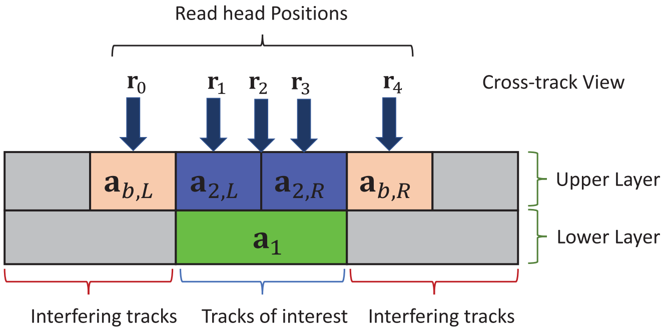

Fig. 2. Cross-track view of the MLMR system.

interest on the upper layer. The two boundary tracks on the left and right sides of $\mathbf{a}_{2,L}$ and $\mathbf{a}_{2,R}$ are denoted by $\mathbf{a}_{b,L}$ and $\mathbf{a}_{b,R}$, respectively. For the lower layer, the written bit sequence $\mathbf{a}_1$ is the track of interest. Readings are obtained at track positions (relative to $\mathbf{r}_0$, which is centered on track $\mathbf{a}_{b,L}$) of 0, 24, 36, 48, and 72 nm, from left to right, and denoted by $\mathbf{r}_0$ to $\mathbf{r}_4$, respectively.

III. CNN Equalizer-Separator Design

We investigate the CNN-based equalizer-separator to use in MLMR system. In [8], we designed the CNN PR equalizers for 1DMR and TDMR systems. To equalize the readings in the MLMR case, we use two separate equalizers for the upper and lower layers. For the lower layer, a CNN equalizer-separator of 1DMR is used, and for the upper layer, a CNN equalizer-separator of TDMR is used. Fig. 3 represents the block diagram of CNN equalizer-separator for the MLMR system. Readings within a 5 × 17 sliding window comprise input samples to the CNN for detecting the upper layer bits. Since each reader collects two samples per lower layer bit, and to maintain a 17-bit down-track footprint, a rate converter multiplexes these additional readings across-track, resulting in size 10 × 17 lower layer input samples. The 17-bit down-track span was chosen based on autocorrelation plots and tuning experiments for BER. We get most of the significant BER performance gains by increasing the window to 17 bits.

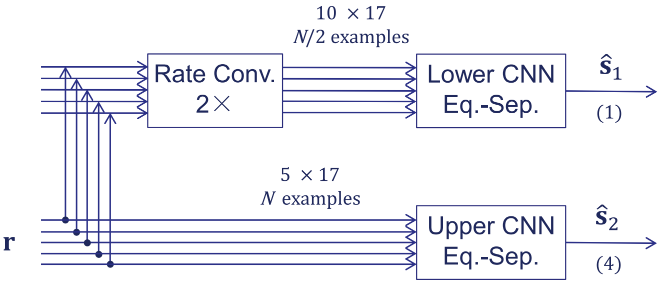

Fig. 3. Block diagram of a CNN equalizer-separator.

A. Nonlinear CNN Equalizer-Separator System

In [8], we designed the CNN equalizer-separator for the bit sequences $\mathbf{a}_1$, $\mathbf{a}_{2,L}$, and $\mathbf{a}_{2,R}$ from input readings. In this work, we input the five sequences of the readings $\mathbf{r}_0, \ldots, \mathbf{r}_4$ to the CNN to generate the equalized waveforms $\hat{\mathbf{s}}_2$ and $\hat{\mathbf{s}}_1$ for the two upper tracks and the lower track.

The commonly used 1-D or 2-D linear PR equalizers minimize the mean-squared error (MSE) between ideal PR signals and the actual output of the equalizer. For the CNN equalizer-separator proposed in the present article, during training, the CNN equalizer iterates with a constrained MSE solver to adjust the PR target masks. Using stochastic gradient descent (SGD) on mini-batches, the equalizer CNN minimizes the average MSE J_{MSE} between its output and the ideal PR waveforms.

The input to the CNN equalizer consists of the readings $\mathbf{r}_0, \ldots, \mathbf{r}_4$ obtained over a sliding window. The outputs are the equalized signals to be fed to the BCJR detector. Training the CNN equalizer is performed as follows [8]. Given a fixed target, the CNN equalizer is trained to minimize J_{MSE}. After the CNN equalizer converges, a constrained MSE solver accepts the fixed CNN equalizer output. Given such output, the solver minimizes the constrained MSE. The constrained MSE solver finds a new MSE-optimal target by adjusting the target coefficients. The new target is used to generate ideal PR signals. Then, the CNN equalizer is retrained on the new ideal PR signal. Iterating between the CNN equalizer and the constrained MSE solver continues until no more significant reductions in J_{MSE} are achieved.

For the lower layer, a rate converter with factor parameter of 2× is used to multiplex the lower layer readings. In this case, the down-track footprint would be 17 bits, same as the upper layer. The CNN receives the five reading sequences and generates the equalized signal $\hat{\mathbf{s}}_1$ for the lower track, using 1×3 PR target.

The 2-D PR equalizer uses a 3×3 PR target $\mathbf{h}_2$ and accepts five adjacent reading sequences to generate the equalized signal $\hat{\mathbf{s}}_2$ for the two upper central tracks.

For the outer tracks, we need to generate the realistic boundaries for the 2-D BCJR detector in the turbo-detection system. In this case, we use a 1-D CNN with the same architecture as the lower layer without a rate converter. To equalize the samples for $\mathbf{a}_{b,L}$, the readings $\mathbf{r}_0, \ldots, \mathbf{r}_3$ with an additional reading obtained by a read-head centered over the track immediately to the left of $\mathbf{a}_1$ are provided for the CNN to produce the equalized waveforms for $\mathbf{a}_{b,L}$. After training, the achieved BER for one block equals to 8.33% in the testing phase. Thus, we add 8.33% BER to the input bits for the outer tracks $\mathbf{a}_{2,L}$ and $\mathbf{a}_{2,R}$ to pass the realistic data to the 2-D BCJR.

B. CNN Equalizer-Separator Architecture

The proposed CNN equalizer-separator for both the upper and lower layers contains 12 layers. These layers consist of one input image layer, three convolutional units, a fully connected (FC) layer, and one output layer. After normalizing the raw data received from the other blocks to have zero mean and unit variance, the system passes them to the input image layer.

Every convolutional unit includes three layers: convolutional layer, batch normalization layer, and activation function layer. The convolutional layer slides the filter over the input data, and the batch normalization layer normalizes the data to speed up network training and reduce sensitivity to the initial conditions (of the filter coefficients and interconnection weights) in the layers. The activation function layer assists the model to converge with greater acceleration. We use the leaky rectified linear unit (ReLU) as the activation function in the CNN equalizer-separators. The output layer is a regression layer. Every convolutional layer has three properties: the filter length, the filter width, and the number of filters which is called the number of channels.

Every node in each FC layer is connected to all the nodes in the previous layer. The output of the last convolutional unit is multiplied by a weight matrix and then a bias vector is added to it to form the output of the FC layer. The last layer is the regression layer, which predicts the accuracy of the model. The regression loss function is 0.5× the MSE between the training label equalized outputs and the CNN equalizer-separator equalized outputs $\hat{\mathbf{s}}_l, l \in \{1, 2\}$. The regression layer is used during training, but not in testing of the CNN. In the testing phase, the final output is generated by the FC layer. Thus, the regression layer is not shown in the CNN architecture figures in this article.

The input image for the equalizers has a length of 17. For the lower layer, since we provide the readings $\mathbf{r}_0, \ldots, \mathbf{r}_4$ and use a rate converter with factor of 2×, thus we pass ten samples into the CNN at a time. Hence, the size of the input image is [10 × 17]. In this layer, the CNN equalizer includes three convolutional units with zero padding. In this case, the size of the input and output images for each convolutional unit is the same. The convolutional layers 1 through 3 have sizes of [3 × 11], [3 × 3], and [3 × 3], respectively. The number of channels for these three convolutional layers is 2, 2, and 1, respectively. The leaky ReLU function with the slope of 0.1 for the negative numbers is used as the activation function. The CNN output $\hat{\mathbf{s}}_1$ is the 1-D equalized waveform of the lower layer. Fig. 4 shows the architecture of CNN equalizer-separator for the lower layer.

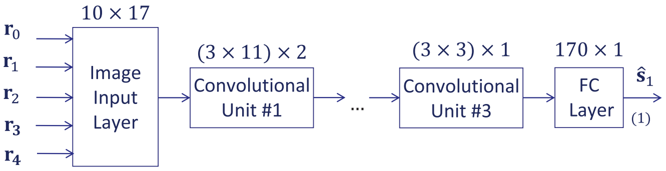

Fig. 4. CNN equalizer-separator architecture for the lower layer.

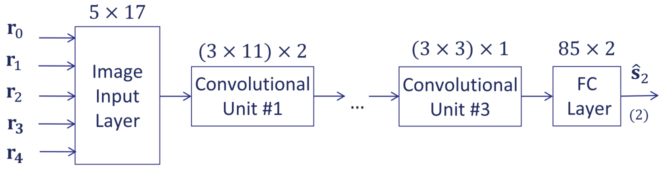

Fig. 5. CNN equalizer-separator architecture for the upper layer.

For the upper layer, a CNN with three convolutional units is used by the equalizer-separator. The architecture of CNN equalizer-separator for the upper layer is represented in Fig. 5. The length of the input image is 17. We pass the readings $\mathbf{r}_0, \ldots, \mathbf{r}_4$ as the input to the CNN, and thus the input image has a size of $[5 \times 17]$. The size of the convolutional layers, the number of channels, and the activation function for this CNN are the same as the CNN equalizer for the lower layer. The CNN generates the equalized waveform $\hat{\mathbf{s}}_2$ for the two upper tracks. These tracks are passed to the 2-D BCJR to detect the bits.

IV. BCJR-LDPC-CNN Turbo Detection System

In [24] and [18], we designed the BCJR-LDPC-CNN turbo detection system for 1DMR and TDMR cases. In these works, we separated the ISI/ITI detection and media noise prediction functions into two detectors in a turbo-equalization structure. The BCJR detectors were provided with the equalized waveforms to estimate the coded bit LLRs. After passing the LLR estimates to a CNN, the media noise was predicted and fed back to BCJR in a turbo structure. The LLR estimates and noise predictions were exchanged iteratively between a BCJR detector, an LDPC channel decoder, and a CNN media noise predictor until BER convergence to a low value. The AD is determined based on the maximum code rate achieved by the LDPC code such that the decoded BER is below 10^{-5}. In this work, we use the BCJR-LDPC-CNN turbo detection

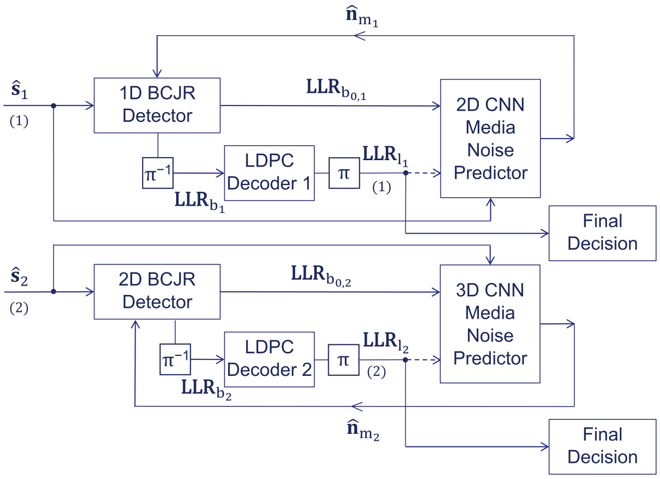

Fig. 6. Block diagram for the MLMR turbo detection system.

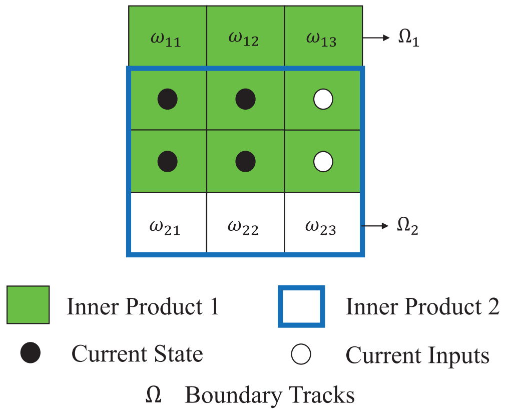

Fig. 7. State input block for the 2-D trellis-based detector with two state bits per track on two tracks.

systems for 1DMR and TDMR cases to investigate a new BCJR-LDPC-CNN turbo-detector for MLMR.

A. MLMR Turbo-Detector

In Fig. 6, the block diagram for the initial version of the MLMR turbo detection system is shown; this initial version has separate CNN noise predictors for the upper and lower layers. Since we have 1-D reading sequence for the lower layer, a BCJR-LDPC-CNN turbo-detector for 1DMR is used to receive the 1-D equalized waveforms from the CNN equalizer-separator and generate the final LLRs. For the upper layer, we deal with 2-D data and detect the coded bits for the two upper tracks. Hence, we use a BCJR-LDPC-CNN detector for TDMR to estimate the LLRs using the 2-D equalized waveforms.

For the lower layer, the equalizer output $\hat{\mathbf{s}}_1$ is passed through a trellis detector (a BCJR detector in this work) to generate the LLR estimates. The 1-D BCJR handles ISI equalization based on the PR target. In this layer, the PR target $\mathbf{h}_1$ has three taps and the ISI channel has length $I = 2$. Hence, the 1-D BCJR has $M = 2^I = 4$ states and eight total branches.

In the first iteration, for the lower layer, the 1-D BCJR LLRs $\mathbf{LLR}_{b_{0,1}}$ and equalized waveform $\hat{\mathbf{s}}_1$ are passed to the CNN to estimate the media noise $\hat{\mathbf{n}}_{m_1}$. The noise $\hat{\mathbf{n}}_{m_1}$ is fed back to the 1-D BCJR to obtain a lower BER.

The LLRs $\mathbf{LLR}_{b_1}$ are input to the channel decoder 1 through the de-interleaver (designated as "π^{-1}" in the block diagram of Fig. 6). The de-interleaver shuffles the coded bits to decorrelate them before they are input to the channel decoder. This shuffling is important because the channel decoder's sum product (SP) algorithm assumes that the incoming LLRs are statistically independent. The decoder can produce extrinsic LLRs relative to $\mathbf{LLR}_{l_1}$ by subtracting the input LLRs (received from the BCJR) from $\mathbf{LLR}_{l_1}$; these extrinsic LLRs can be passed to the CNN media noise predictors if a second iteration is done. We use an irregular repeat accumulate (IRA) LDPC decoder as the channel decoder [25]. Henceforth, we refer to the "IRA decoder" or simply "IRA" to indicate the specific LDPC decoder used in this article. The IRA decoder uses coset decoding since the GSP data bits are randomly distributed. The IRA encoding and decoding is done separately for each track; hence, different code rates can be used on different tracks. The channel decoder produces the output LLRs, and then scales and magnitude limits them. After limiting and scaling, they are passed through an interleaver (indicated by "π" in the block diagram of Fig. 6) to reorder the LLRs to be consistent with the order of $\mathbf{LLR}_b$. For the lower layer, the final LLRs $\mathbf{LLR}_{l_1}$ are generated by passing the LLRs of LDPC decoder 1 through an interleaver at the end of each turbo-iteration.

For the upper layer, we use a BCJR-LDPC-CNN turbo-detector for TDMR. In this system, we separate trellis-based ISI/ITI detection and CNN-based media noise prediction.

The 2-D BCJR is a joint ISI/ITI equalizer; the state input block, shown in Fig. 7, has three rows because the ITI typically extends over one adjacent track on either side. In 2-D BCJR, the processing of two tracks is done simultaneously to handle the ITI from a 3 × 3 PR target mask since the two upper tracks are affected by the ITI from its two neighboring tracks. Fig. 7 represents the state input block for 2-D BCJR over two tracks. The two 3 × 3 inner products (between the 3 × 3 PR mask $\mathbf{h}_2$ and the bits indicated by either shading or bold outline) shown in Fig. 7 are used as the conditional means for the conditional channel probability density functions (PDFs) of the two "current inputs" bits shown in Fig. 7. These conditional channel PDFs are independent factors in the gamma probabilities for the 2-D BCJR algorithm [26]. The 2-D BCJR trellis detector performs ISI/ITI equalization on filtered input $\hat{\mathbf{s}}_2$ and generates LLR outputs. The PR target $\mathbf{h}_2$ is a size of 3 × 3; hence, the system has two state bits per track, and the 2-D BCJR state input window is 2 × 3, and the trellis has $2^{2\times 2} = 16$ states.

The flow of exchanging LLR estimates and media noise predictions in the turbo-detector in the upper layer is similar to the lower layer. In the first iteration, the 2-D BCJR LLRs $\mathbf{LLR}_{b_0,2}$ and $\hat{\mathbf{s}}_2$ are passed to the CNN to estimate the media noise $\hat{\mathbf{n}}_{m_2}$. The noise $\hat{\mathbf{n}}_{m_2}$ is fed back to the 2-D BCJR to obtain a lower BER. The LLRs $\mathbf{LLR}_{b_2}$ are input to the LDPC decoder 2 through the de-interleaver. The LLRs of the channel decoder 2 are input to an interleaver to produce the final LLRs $\mathbf{LLR}_{l_2}$.

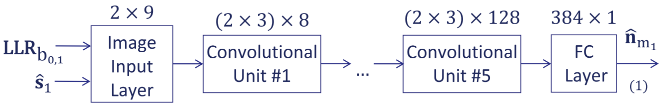

Fig. 8. 2-D CNN architecture of separate media noise predictor for the lower layer.

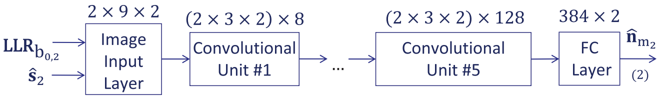

Fig. 9. 3-D CNN architecture of separate media noise predictor for the upper layer.

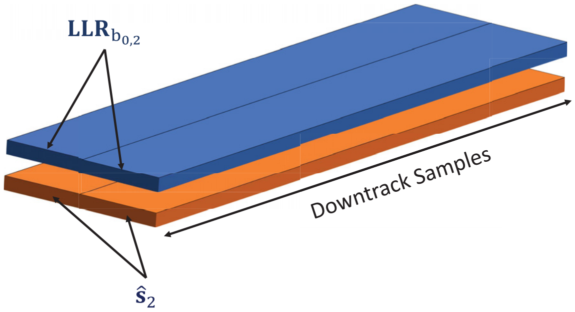

Fig. 10. 3-D input image for the separate CNN noise predictor of the upper layer.

As explained in [18], for the second iteration, extrinsic decoder LLRs based on $\mathbf{LLR}_{l_k}$ can be passed as inputs to the CNN noise predictor instead of $\mathbf{LLR}_{b_k}$. In this article, we do only one iteration of LDPC decoding; gains due to a second decoding iteration will be investigated in future work.

V. CNN Noise Predictor Architecture

We investigate two architectures for the CNN media noise predictor in MLMR with two layers. In the first architecture, we use separate CNNs for the lower and upper layers. A 2-D CNN for the 1DMR system and a 3-D CNN for the TDMR system are considered as the architectures for the lower and upper layers, respectively. In the second architecture, we use a multilayer CNN noise predictor for both the upper and lower layers. The multilayer CNN receives the LLR estimates and equalized waveforms from both the layers and jointly predicts the media noise for both the layers. The multilayer CNN can account for both residual ILI and interlayer media noise by exploiting correlations between the LLRs and equalized sampled data from the upper and lower layers.

A. Separate CNN Architecture

The CNN architecture extracts correlations among the data and exploits them to obtain the information and features. For 1DMR, we stack the data as 2-D input images, and hence, we design 2-D CNNs that use banks of 2-D finite impulse response (FIR) filters to predict media noise. In the TDMR detector, we stack the 2-D input images for three tracks to make a 3-D input image. Therefore, for processing the 3-D input image, we design a 3-D CNN architecture that uses banks of 3-D FIR filters to estimate the media noise for the TDMR system.

The CNNs process their input data in a sliding block manner. For 1DMR, to estimate the kth media noise sample $\hat{n}_{m_k}$, the lowest input layer of the 2-D CNN accepts a block of N_i 1-D BCJR output LLRs $\mathbf{LLR}_{b_{0,1_k}}$ and N_i equalized readings $\hat{\mathbf{s}}_{1_k}$ where N_i is an odd number, and the kth noise estimate corresponds to the middle element of the N_i elements in each block; in this article, we set $N_i = 9$. To estimate the $(k+1)$th media noise sample, each of the input data blocks is shifted by exactly one sample into the future.

The TDMR system processes two tracks at a time. Hence, for TDMR, each of the above-described 1-D input blocks of size $[1 \times N_i]$ becomes a $[2 \times N_i]$ input block.

1) 2-D CNN Noise Predictor Architecture for 1DMR: The 2-D CNN architecture for the first iteration of the lower layer 1DMR system is shown in Fig. 8. The 2-D CNN has 18 layers including one input image layer, five convolutional units, and one output layer. The 2-D input image layer is of size 18 and includes two rows consisting of nine samples from each of the two input blocks $\mathbf{LLR}_{b_0,1}$ and $\hat{\mathbf{s}}_1$. Organizing the 1-D input blocks into a 2-D array in this manner induces 2-D spatial correlation between the blocks. We exploit this spatial correlation using trained 2-D convolutional filters on all the CNN layers. In CNNs designed for 1DMR, all the convolutional layers use filters of size $[2 \times 3]$. The number of channels in units 1 through 5 is equal to 8, 16, 32, 64, and 128, respectively, for the first iteration. ReLU is used as the activation function in this architecture.

2) 3-D CNN Noise Predictor Architecture for TDMR: For the upper layer TDMR detector, we design the 3-D CNN to predict the media noise. Fig. 9 represents the 3-D CNN noise predictor architecture. The numbers of layers for the 3-D CNN of TDMR system are the same as 2-D CNN for the 1DMR system. Each of the two input layer 2-D images contains $2 \times 9 = 18$ samples, and thus the size of the CNN's 3-D input layer is $2 \times 18 = 36$ samples. The 3-D input image contains the 2-D input images for the two tracks. By putting the two 2-D input images together, we can extract the information from the 3-D spatial correlation between the input images and the correlation between the two tracks of each block simultaneously. In Fig. 10, stacking of the 2-D input images (with two tracks each) into a 3-D input image is shown. In the designed 3-D CNNs for TDMR, the convolutional layers have the filters with the size of $[2 \times 3 \times 2]$. Similar to the 2-D CNN architecture, the number of channels through convolutional units 1 to 5 is 8, 16, 32, 64, and 128. For the activation function, we use ReLU the same as 2-D CNN media noise predictor.

B. Multilayer CNN Noise Predictor

For the multilayer CNN noise predictor, we investigate a 3-D CNN to estimate the media noise for both the upper and lower layers simultaneously. Fig. 11 represents the block diagram for the MLMR turbo detection system using the

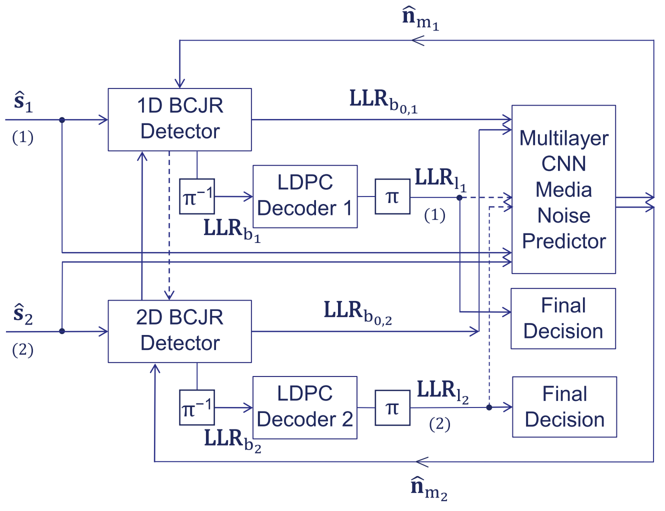

Fig. 11. Block diagram for MLMR turbo-detector using the multilayer CNN media noise predictor.

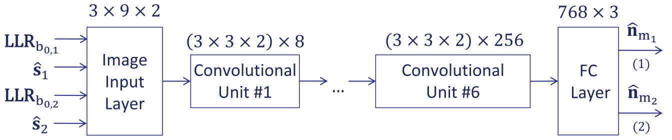

Fig. 12. 3-D CNN architecture of multilayer media noise predictor for MLMR.

multilayer media noise predictor. In this architecture, the estimation of LLRs and the equalized waveforms for both the upper and lower layers are passed to the CNN.

The 3-D CNN architecture for the multilayer CNN noise predictor is shown in Fig. 12. This 3-D CNN includes 21 layers and six convolutional units. We store the LLR estimates and equalized waveforms for the lower track adjacent to the corresponding data for the upper tracks. Since BL for the lower track is twice the upper tracks' BL, we duplicate the bits for the lower track to have the same number of samples as each upper track. Therefore, for the output of the lower layer, we average two consecutive output bits to have the same down-track length as the original lower layer's. Since for each track we have 18 samples for the input image, the size of the input image would be $3 \times 18 = 54$. The data stacking of the upper and lower tracks to make the 3-D input image for the multilayer CNN noise predictor is represented in Fig. 13. By storing the lower track's data adjacent to the upper tracks', we could extract the correlations between the two layers. In the proposed CNN, the size of the filters for all convolutional units is $[3 \times 3 \times 2]$. The number of channels through unit 1 to 6 increases from 8 to 256 by a factor of 2×.

VI. LLR Exchange Between Layers

Due to the residual ILI after the equalizer-separator, we hypothesize that exchanging the LLRs between 1-D BCJR and 2-D BCJR in the lower and upper layers could reduce the residual ILI and improve the quality of the output LLRs of both BCJRs. The solid and dotted lines in Fig. 11 show the LLR exchange between the two layers. Each BCJR could reduce the ILI by assuming some weights for mapping the corresponding coded bits from the other layer to the current layer. For each layer BCJR, the mapping weights $\mathbf{w}$ are obtained by solving the following least squared error (LSE) problem:

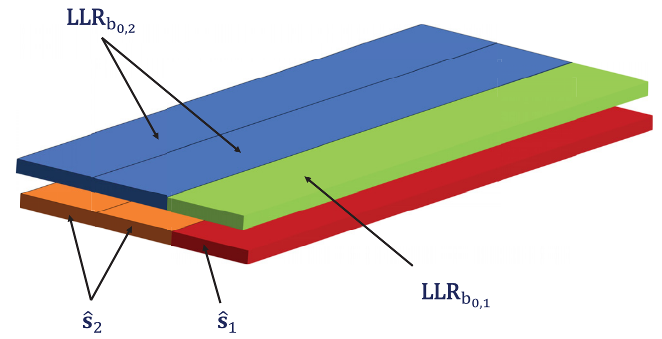

Fig. 13. 3-D input image for the multilayer CNN noise predictor.

$$\min_{\mathbf{w}} \|\hat{\mathbf{s}} - \mathbf{h} * \mathbf{u} - \mathbf{w} * \mathbf{L}_d\|^2 \tag{2}$$

where $\mathbf{L}_d$ is the hard-thresholded LLRs (to ±1) passed from the other layer's BCJR.

The solid line in Fig.11 shows the LLRs flowing from the 2-D BCJR in the upper layer to 1-D BCJR in the lower layer. Since for the lower layer the PR target includes three taps, we consider three samples in this layer and 12 associated samples in the upper layer. For each track in the upper layer, we map six samples to the lower. Hence, by solving (2), the corresponding weights $\mathbf{w}_1$ and $\mathbf{w}_2$ for the two tracks of interest in the upper layer are obtained as following:

$$\mathbf{w}^{\mathsf{T}} = \begin{bmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{bmatrix}^{\mathsf{T}} = \begin{bmatrix} -0.0023 & -0.0056 \\ 0.0247 & 0.0383 \\ 0.0675 & 0.0899 \\ 0.0556 & 0.0546 \\ 0.0071 & 0.0046 \\ -0.0039 & -0.0034 \end{bmatrix}. \tag{3}$$

Using these weights to map the upper layer's signals to the lower layer, the lower layer's MSE [computed via (2)] decreases by 20.10%. This MSE reduction leads to the AD improvement for the lower layer and consequently for MLMR system which will be discussed.

After passing the upper layer's LLRs to the 1-D BCJR in the lower layer, we use the row-column soft decision feedback algorithm (IRCSDFA) for the bit detection in the lower layer's 1-D BCJR as follows [26]. Given input vector $\mathbf{u}_k = [u_k, u_{k-1}, u_{k-2}]$, the current state $S_k = (u_{k-1}, u_{k-2})$. We denote the current equalized sample as $\hat{s}_{1_k}$, and branch bits $\mathbf{i} = [i_0, \ldots, i_{n_b-1}]$, $i_m \in \{-1, +1\}$ where n_b is the number of input per trellis stage; n_b is 1 in our work. The modified conditional channel probability density function (pdf) $p'(.)$ sums over the values of inner products $\mathbf{w} * \mathbf{L}_d$ associated with the 12 corresponding upper level bits, for the state transition $s' \to s$

$$p'\left(\hat{s}_{1_k} | u_k = i_0, S_k = s, S_{k-1} = s'\right) = P\left(\hat{s}_{1_k} | u_k, s, s', \mu_k + \mu(\mathbf{L}_d)\right) \quad (4)$$

where

$$\mu_k = \mathbf{h} * \mathbf{u}_k = h_0 u_k + h_1 u_{k-1} + h_2 u_{k-2} \quad (5)$$

$$\mu(\mathbf{L}_d) = \sum_{\mathbf{L}_d} P(\mathbf{L}_d) \times (\mathbf{w} * \mathbf{L}_d) \quad (6)$$

$\mathbf{w}$ is defined in (3), $\mathbf{L}_d$ is the vector of 12 upper layer bits above lower layer bits $\mathbf{u}_k$, and the row probabilities

$$P(\mathbf{L}_d = [l_0, l_1, \ldots, l_{11}]) = \prod_{j=0}^{11} P\left(l_j\right) \quad (7)$$

such that $P(l_j)$ are the bit probabilities derived from the LLRs passed from the upper layer to the lower layer. In this case, the 1-D BCJR in the lower layer can use the LLRs from the upper layer to reduce ILI and detect the coded bits with lower BER.

The dotted line in Fig. 11 represents passing the 1-D BCJR LLRs from the lower layer to the 2-D BCJR in the upper layer. Since the number of samples in the upper layer is twice that in the lower layer, we duplicate the LLRs coming from the lower layer. The three-input three-output PR target leads us to assume a window of weights with a size of 3 for each track to solve the LSE problem in (2). However, the obtained weights are too small and could not reduce the MSE. As future work, we will investigate new techniques to use the lower layer's LLRs to improve the AD of the upper layer.

VII. Simulation Results

This section presents the simulation results for the BCJR-LDPC-CNN turbo-detector on an MLMR channel. The simulations are performed on the GSP datasets with TPs 24 and 48 nm for the upper and lower layers. We provide the results for both linear and CNN-based PR equalization before the turbo-detector.

The BCJRs initially assume that the media noise is zero and compute an initial set of output LLRs $\mathbf{LLR}_{b_{0,1}}$ and $\mathbf{LLR}_{b_{0,2}}$. The CNN media noise predictors are provided with the LLR probabilities. The estimated bit and the estimation's reliability can be determined by the LLR probabilities. In [24], it is experimentally shown that CNN provided with LLR probabilities performs better compared with when it uses the signed LLR values. This might be due to the non-linear scale inherent in the LLRs.

A. Datasets

We use the GSP waveform dataset which has 10 nm BL and 24 nm TP for the upper layer and 20 nm BL and 48 nm TP for the lower layer. The upper layer in each block contains $6 \times N_u$ input bits, where $N_u = 82\,412$. For this layer, the simulation results consider the two middle tracks as the tracks of interest and assume their side tracks as the boundary tracks. For the lower layer, each block includes $3 \times N_l$ input bits, where $N_l = 41\,206$. The track of interest in this layer is the middle track.

For all CNN-based simulations, we use 59, 1, and 40 blocks as the training, validation, and test datasets, respectively for both CNN equalizer-separator and BCJR-LDPC-CNN detector. For 1-D PDNP, we use 1 and 40 blocks for the training and testing processes, respectively.

B. Simulation Parameters

The iterations between the BCJR, IRA, and CNN are implemented. In this work, the weight and magnitude limit threshold for the BCJR output LLRs are set to 0.1 and 60 for the 1DMR system and 0.7 and 100 for the TDMR system, respectively. After scaling and limiting LLRs $\mathbf{LLR}_{b_1}$ and $\mathbf{LLR}_{b_2}$, they are passed to the de-interleaver to shuffle the code bits. The minimum internal IRA code iteration is set to 200 and the maximum iteration is set to 400. The weight and magnitude limit threshold after the decoder are considered to scale and limit the decoder output LLRs $\mathbf{LLR}_l$. In the proposed system, the weight and limit threshold for the decoder output LLRs equal 0.1 and 100 for the 1DMR system and 0.5 and 100 for the TDMR system.

We report the results in terms of AD which is determined based on the achieved code rate which is the highest code rate that yields a final decoded BER equal to or less than 10^{-5}. Due to the difficulty in designing IRA codes with different rates, at first, a few IRA codes with some specific code rates are designed. To achieve the highest code rate of a new system, we select one of the designed IRA codes as the base code, and by puncturing process we could reach to the higher code rate. The rate of the base code is called "base rate." The puncturing scheme in [27] is used to simulate puncturing bits written to an HDD. The AD is obtained by

$$\text{Areal density} = \text{achieved-code-rate}/(\text{BL} \times \text{TP}). \quad (8)$$

The code rate for the MLMR system is computed based on the code rates of the upper layer and lower layer as follows:

$$\text{code-rate}_{\text{MLMR}} = \text{code-rate}_{\text{Upper}} + (1/4) \times \text{code-rate}_{\text{Lower}}. \quad (9)$$

Table I summarizes the results for the MLMR detectors. The raw channel BER for the lower layer is reported for the detected bits on the input readings of the central single-track $\mathbf{a}_1$. For the upper layer, the raw channel BER is the average detected errors on the two central tracks $\mathbf{a}_{2,L}$ and $\mathbf{a}_{2,R}$. In addition to the BER, a 95% confidence upper bound on the BER is provided in parentheses in the last columns of Table I. In the case of non-zero errors, the BER upper bound is computed as

$$I_p(x+1, N_{tcb} - x) = \gamma \quad (10)$$

where I_p indicates the beta distribution with parameters $x+1$ and $N_{tcb} - x$ for γ-quantile, x is the number of errors, N_{tcb} is the total number of transmitted coded bits, and γ is the confidence threshold which we set to 0.95. In the case of zero error count, the BER upper bound in (10) simplifies to $3/N_{tcb}$ [28].

TABLE I

AREAL DENSITY COMPARISON FOR MLMR. NP MEANS NOISE PREDICTION

Method/Layer	**TP** (nm)	**Raw Channel BER**	**AD** (Tb/in^2)	**Code Rate**	**Decoded BER**	**AD Gain of Best Result Over (%)**
1-D PDNP, Reference, Lin.-Eq.	24	0.0560	2.3460	0.8727	0 (9.10e-7)	13.18
1-D PDNP, Upper, CNN-Eq.	24	0.3577	2.4900	0.7286	0 (9.10e-7)	6.63
1-D PDNP, Lower, CNN-Eq.	48	0.1094		0.7909	0 (1.82e-6)	
Separate CNN NP, Reference, CNN-Eq.	24	0.0560	2.3852	0.8873	0 (9.10e-7)	11.32
Separate CNN NP, Upper, CNN-Eq.	24	0.3577	2.5147	0.7306	0 (9.10e-7)	5.58
Separate CNN NP, Lower, CNN-Eq.	48	0.1094		0.8195	0 (1.82e-6)	
Multilayer CNN NP, CNN-Eq.	24/48	0.3577/0.1094	2.6506	0.9860	0 (1.82e-6)	0.17
Multilayer CNN NP, Exchanged LLR, CNN-Eq.	24/48	0.3577/0.1094	**2.6551**	0.9877	0 (1.82e-6)	-
Multilayer CNN NP, Exchanged LLR, CNN-Eq., SNR 20 dB	24/48	0.3624/0.1094	2.5296	0.9410	0 (1.82e-6)	4.96

C. Discussion of Simulation Results

Table I summarizes the results for the GSP dataset for MLMR. The table shows the comparison of the AD performance of the proposed BCJR-LDPC-CNN detectors with 1-D PDNP. "1-D PDNP, Lin.-Eq." uses the equalized waveforms from a linear minimum mean-squared error (MMSE) equalizer using a 1-D PR target. "1-D PDNP, CNN-Eq." uses the same PR target $\mathbf{h}_1$ as the BCJR-LDPC-CNN method for the lower layer and two 1-D PR targets for each of the two upper tracks. 1-D PDNP has 128 trellis states, corresponding to $I = 2$, $L = 4$, and $\Delta = 1$, where I is the ISI channel length, L is the predictor memory, and Δ is the predictor look-ahead. The pattern vector length of $I + 1 + L + \Delta = 8$ bits of 1-D PDNP is comparable to the channel inputs $\hat{\mathbf{s}}_1$ and $\hat{\mathbf{s}}_2$ length of nine sample bits for the CNN.

As a reference, we evaluate a one-layer TDMR system (without lower layer interference) with TP 24 nm and BL 10 nm using the upper layer's BCJR-LDPC-CNN architecture. As a second reference, we evaluate the performance of the one-layer TDMR system (without lower layer interference) on a system that includes a 1-D linear PR equalizer and BCJR/1-D PDNP detector for each of the two upper layer tracks. The 1-D linear PR equalizer is designed according to the method in [29].

The base code rate is 0.6030 for all the simulations in Table I. By puncturing the base code, we could achieve the highest code rate for every detector and layer. The ADs for each detector are computed using (8).

Based on (9), the achieved code rate for the MLMR system is obtained. For instance, the highest code rate for the separate CNN noise predictor using CNN equalizer-separator for the upper and lower layers shown as "Separate CNN NP, Upper, CNN-Eq." and "Separate CNN NP, Lower, CNN-Eq." are 0.7306 and 0.8195, respectively. Thus, the achieved code rate for the MLMR system using separate CNN noise predictor and CNN equalizer-separator equals to $0.7306 + (1/4) \times 0.8195 = 0.9354$. The achieved AD for this case is 2.5147 based on (8).

The last column in Table I represents the AD gain of the best result over the detector in each line. The best result in the simulations is for the multilayer CNN noise predictor using the CNN equalizer-separator with the LLR exchange between the layers shown as "Multilayer CNN NP, Exchanged LLR, CNN-Eq." which has 2.6551 Tb/in^2 density. As an instance for the last column, the AD for the 1-D PDNP using the CNN equalizer-separator (shown in the second and third lines of Table I) has 2.4900 Tb/in^2 density. Thus, the gain of best result over this detector would be 6.63% represented in the last column. The best result for the proposed method has 11.32% AD gain over "Separate CNN NP, Reference, CNN-Eq." as the first reference and has AD gain of 13.18% over the "1-D PDNP, Reference, Lin-Eq." as the second reference. Both the references include only upper layer as a TDMR system.

Using the multilayer CNN architecture instead of the separate CNN architecture, the AD increases from 2.5147 to 2.6506 Tb/in^2 which is 5.40% gain. Also, passing the LLRs from the upper layer's BCJR to the lower layer's BCJR increases 0.73% AD for the lower layer and results in a density gain of 0.17% for MLMR system over the case without LLRs passing. Hence, combining these two methods leads to the best result having 5.58% gain over the separate CNN noise predictor using the CNN equalizer-separator.

The best achieved AD of 2.6551 Tb/in^2 in Table I is greater than the value of 1.0616 achieved in [8], as expected since the TP in the present article is 1/2 that in [8]. Future work will consider even smaller TPs and possibly also a higher area ratio between the upper and lower level bits; both will result in potentially higher information ADs.

The GSP data contain no read-head electronic AWGN, i.e., $n_e(k) = 0$ in (1). The rows without label "SNR 20 dB" report results for this case. The row which includes the label "SNR 20 dB" reports the result when non-zero AWGN n_e at an SNR of 20 dB is added to the superposition signals corresponding to the upper layer readings $\mathbf{r}_0$, $\mathbf{r}_1$, $\mathbf{r}_3$, and $\mathbf{r}_4$. The SNR is computed as

$$\mathrm{SNR} = 10 \log_{10}\left(\frac{\mathbb{E}\left[\|\mathbf{r}_0\|^2 + \|\mathbf{r}_1\|^2 + \|\mathbf{r}_3\|^2 + \|\mathbf{r}_4\|^2\right]}{\sigma_e^2}\right) \quad (11)$$

where σ_e^2 indicates the AWGN variance. This SNR is computed based on all the coded bits of each track. In (11), the reading $\mathbf{r}_2$ corresponding to the lower layer is excluded

TABLE II
DETECTOR COMPLEXITY FOR THE MLMR

Method/Layer	mul/div	add/sub	exp/log
1-D PDNP	16,266	15,626	257
Separate CNN NP, Upper	238,779	215,687	66
Separate CNN NP, Lower	157,140	155,957	18
Multilayer CNN NP	1,286,103	1,188,869	50

since the AWGN is considered only for the upper layer. The simulation results for the added AWGN at 20 dB SNR are shown in the last line of Table I; they show that best result with zero electronic noise has an AD gain of a reasonably small amount (4.96%) over the same method with 20 dB SNR for AWGN.

D. Computational Complexity Comparison

The computational complexity (per bit) figures for the MLMR detectors are represented in Table II. The reported numbers are computed without considering the equalizer. The 1-D PDNP which is the baseline has the lowest complexity among the evaluated detectors. The complexity of the separate CNN noise predictor for the upper and lower layers is shown as "Separate CNN NP, Upper" and "Separate CNN NP, Lower," respectively. The upper layer's detector is more complex than the lower layer's, since we use a 3-D CNN for the upper layer; on the other hand, we use a 2-D CNN for the lower layer. The multilayer CNN noise predictor shown as "Multilayer CNN NP" has the highest complexity due to the processing of the data for upper and lower layers simultaneously; specifically, the input layer size and FIR filter sizes are larger for the multilayer CNN compared with the separate CNNs, as per Figs. 8, 9, and 12. Both the separate and multilayer CNN architectures have more complexity compared with 1-D PDNP. As future work, we will investigate trade offs between the complexity and the performance for the proposed CNN noise predictors in this work.

VIII. CONCLUSION

This article presents CNN-based methods for equalization and detection of MLMR signals. Combining the CNN equalizer-separator with the BCJR-LDPC-CNN media noise predictor turbo-detection system enables us to remove ITI and ILI and improve the AD of the system. In this work, we investigate the proposed methods on a two-layer MLMR. For the upper layer, the CNN equalizer-separator designs a 2-D PR target to equalize the readings and provides them to turbo-detector for TDMR which uses a 2-D BCJR to detect the coded bits for the two tracks of interest. For the lower layer, the equalizer-separator optimizes a 1-D PR target to equalize the 1-D sample readings. The equalized waveforms are passed to a turbo-detection system for 1DMR which uses a 1-D BCJR. In this case, equalizer-separator focuses on ILI and the BCJRs handle the ITI. We investigate two architectures for the CNN noise predictor in the turbo-detection system. In the first architecture, the 2-D and 3-D CNN are designed for the lower and upper layers separately. In the second one, a multilayer CNN predicts the media noise for both the layers simultaneously, to take into account inter-layer media noise. Also, the LLRs from the upper layer are passed to the lower layer to reduce the residual ILI. According to the simulations, the proposed method has AD gain of 11.32% over the same architecture which includes only the upper layer, 13.18% AD gain over the 1-D PDNP using linear equalizer applied only on the upper layer, and 6.63% gain compared with 1-D PDNP on individual tracks of both the upper and lower layers using the CNN equalizer-separator. As future work, we will investigate the second pass of LLRs to the CNN noise predictor in the second iteration of the turbo-detection system. We will study the second pass of LLRs from the upper layer's BCJR to lower layer's, after noise correction is done for the second pass of BCJR in the upper layer. Also, we will consider further investigation of performance/complexity trade offs in the CNN architectures.

ACKNOWLEDGMENT

This work was supported in part by the Advanced Storage Research Consortium and in part by the United States National Science Foundation under Grant CCF-1817083.

REFERENCES

[1] S. J. Greaves, K. S. Chan, and Y. Kanai, "Areal density capability of dual-structure media for microwave-assisted magnetic recording," *IEEE Trans. Magn.*, vol. 55, no. 12, pp. 1–9, Dec. 2019.

[2] S. J. Greaves, K. S. Chan, and Y. Kanai, "Optimization of dual-structure recording media for microwave-assisted magnetic recording," *IEEE Trans. Magn.*, vol. 55, no. 7, pp. 1–5, Jul. 2019.

[3] K. S. Chan, S. Greaves, and S. Rahardja, "Optimization of the 3-D-MAMR media stack," *IEEE Trans. Magn.*, vol. 55, no. 9, pp. 1–5, Sep. 2019.

[4] K. S. Chan, R. Wood, and S. Rahardja, "Maximum likelihood detection for 3-D-MAMR," *IEEE Trans. Magn.*, vol. 55, no. 12, pp. 1–9, Dec. 2019.

[5] K. S. Chan, A. Aboutaleb, K. Sivakumar, B. Belzer, R. Wood, and S. Rahardja, "Data recovery for multilayer magnetic recording," *IEEE Trans. Magn.*, vol. 55, no. 12, pp. 1–16, Dec. 2019.

[6] S. Yoon and E. Hwang, "Joint multi-level readback for array-reader-based interlaced magnetic recording," *IEEE Trans. Magn.*, vol. 57, no. 3, pp. 1–5, Mar. 2021.

[7] Y. Li, Y. Wang, Y. Xu, L. Chen, Y. Wen, and P. Li, "Multitrack detection with 2D iterative soft estimate aided neural network equalizer for heat assisted interlaced magnetic recording," *IEEE Trans. Magn.*, vol. 57, no. 3, pp. 1–8, Mar. 2021.

[8] A. Aboutaleb *et al.*, "Deep neural network-based detection and partial response equalization for multilayer magnetic recording," *IEEE Trans. Magn.*, vol. 57, no. 3, pp. 1–12, Mar. 2021.

[9] A. Aboutaleb *et al.*, "A perspective on deep neural network-based detection for multilayer magnetic recording," *Appl. Phys. Lett.*, vol. 119, no. 1, pp. 1–12, Jul. 2021.

[10] R. Wood, M. Williams, A. Kavcic, and J. Miles, "The feasibility of magnetic recording at 10 Terabits per square inch on conventional media," *IEEE Trans. Magn.*, vol. 45, no. 2, pp. 917–923, Feb. 2009.

[11] J. Yao, E. Hwang, B. V. K. V. Kumar, and G. Mathew, "Two-track joint detection for two-dimensional magnetic recording (TDMR)," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 418–424.

[12] Y. Wang and B. V. K. V. Kumar, "Multi-track joint detection for shingled magnetic recording on bit patterned media with 2-D sectors," *IEEE Trans. Magn.*, vol. 52, no. 7, pp. 1–7, Jul. 2016.

[13] Y. Wang and B. V. K. V. Kumar, "Micromagnetics-based analysis of multi-track detection with simplified 2-D write precompensation on shingled magnetic recording," *IEEE Trans. Magn.*, vol. 52, no. 9, pp. 1–11, Sep. 2016.

[14] 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.

[15] K. Luo *et al.*, "A study on block-based neural network equalization in TDMR system with LDPC coding," *IEEE Trans. Magn.*, vol. 55, no. 11, pp. 1–5, Nov. 2019.

[16] 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.

[17] 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.

[18] A. Sayyafan *et al.*, "Deep neural network media noise predictor turbo-detection system for 1-D and 2-D high-density magnetic recording," *IEEE Trans. Magn.*, vol. 57, no. 3, pp. 1–13, Mar. 2021.

[19] J. Shen, B. J. Belzer, K. Sivakumar, K. S. Chan, and A. James, "Convolutional neural network based symbol detector for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 57, no. 3, pp. 1–5, Mar. 2021.

[20] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–444, May 2015.

[21] 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.

[22] M. Nishikawa, Y. Nakamura, Y. Kanai, H. Osawa, and Y. Okamoto, "Improvement of iterative decoding with LLR modulator by neural network using magnetic transition information in SMR system," *IEEE Trans. Magn.*, vol. 57, no. 2, pp. 1–5, Feb. 2021.

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

[24] 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.

[25] H. Jin, A. Khandekar, and R. McEliece, "Irregular repeat-accumulate codes," in *Proc. 2nd Int. Symp. Turbo Codes Rel. Topics*, Brest, France, Sep. 2000, pp. 1–8.

[26] T. Cheng, B. J. Belzer, and K. Sivakumar, "Row-column soft-decision feedback algorithm for two-dimensional intersymbol interference," *IEEE Signal Process. Lett.*, vol. 14, no. 7, pp. 433–436, Jul. 2007.

[27] 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.

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

[29] C. K. Matcha and S. G. Srinivasa, "Generalized partial response equalization and data-dependent noise predictive signal detection over media models for TDMR," *IEEE Trans. Magn.*, vol. 51, no. 10, pp. 1–15, Oct. 2015.