Deep Neural Network Based Media Noise Predictors for Use in High-Density Magnetic Recording Turbo-Detectors

Amirhossein Sayyafan[1], Benjamin J. Belzer[1], Krishnamoorthy Sivakumar[1], Jinlu Shen[1], Kheong Sann Chan[2], and Ashish James[3]
[1]School of Electrical Engineering and Computer Science, Washington State University, Pullman, WA 99164-2752 USA
[2]Nanjing Institute of Technology, Nanjing 211167, China
[3]Institute for Infocomm Research (I2R), A*STAR, Singapore

This paper presents a combined Bahl-Cocke-Jelinek-Raviv (BCJR) and deep neural network (DNN) turbo-detection architecture for one dimensional (1D) hard disk drive (HDD) magnetic recording. Simulated HDD readings based on a grain flipping probabilistic (GFP) model are input to a linear filter equalizer with a 1D partial response (PR) target. The equalizer output is provided to the BCJR detector in order to minimize the intersymbol interference (ISI) due to the PR mask. The BCJR detector's log-likelihood-ratio (LLR) outputs (along with the linear equalizer outputs) are then input to the DNN detector, which estimates the signal dependent media noise. The media noise estimate is then fed back to the BCJR detector in an iterative manner. Several DNN media noise estimation architectures based on fully connected (FC) and convolutional neural networks (CNNs) are investigated. For GFP data at 48 nm track pitch and 11 nm bit length the CNN-based BCJR-DNN turbo detector reduces the detector BER by $0.334\times$ and the per bit computational time by $0.731\times$ compared to a BCJR detector that incorporates 1D pattern-dependent noise prediction (PDNP). The proposed BCJR-DNN turbo detection architecture can be generalized for two-dimensional magnetic recording (TDMR).

***Index Terms*—Deep neural network, convolutional neural network, Bahl-Cocke-Jelinek-Raviv(BCJR) detectors, magnetic recording, turbo-detectors.**

I. Introduction

TRellis based detection with pattern dependent noise prediction (PDNP) has become standard practice in the HDD industry. In typical single track signal processing, the received samples from the read head are filtered by a linear equalizer with a 1D partial response (PR) target $\mathbf{h}$, giving an effective channel model of $\mathbf{y} = \mathbf{h} * \mathbf{u} + \mathbf{n}_m + \mathbf{n}_e$, where $\mathbf{u}$ are the coded bits on the track, * indicates 1D convolution, $\mathbf{n}_m$ is media noise, $\mathbf{n}_e$ is reader electronics AWGN, and the effective ISI channel length $I = \text{length}(\mathbf{h}) - 1$. The term $\mathbf{n}_m$ models signal dependent noise due to, e.g., magnetic grains intersected by bit boundaries, which can influence two or more bit readback values. The equalizer output $\mathbf{y}$ flows into a trellis-based (Viterbi [1] or BCJR [2]) detector that employs a super-trellis based on the effective ISI channel and a 1D pattern dependent noise prediction (1D PDNP) algorithm. PDNP is based on an Lth order trained autoregressive media noise model $\tilde{n}_{m_k}$: $\tilde{n}_{m_k}(\mathbf{u}_k) = \sum_{i=1}^{L} a_i(\mathbf{u}_k) n_{m_{k-i}}(\mathbf{u}_k) + e_k(\mathbf{u}_k)$, where the a_i are the auto-regressive coefficients, and the model error e_k is assumed to be uncorrelated Gaussian noise that depends on the coded bit pattern vector $\mathbf{u}_k = [u_{k+\Delta}, \ldots, u_k, \ldots, u_{k-(I+L)}]$ [3], [4]. The number of super-trellis states for N_t-track PDNP detection is $2^{N_t(I+L+\Delta)}$, with typical values $I \leq 3$, $L \leq 4$ and $\Delta \leq 1$. The L model coefficients $a_i(\mathbf{u}_k)$ for each pattern vector $\mathbf{u}_k$ are trained and computed offline using the normal equation. The trellis detector sends soft coded bit estimates to a channel decoder to determine the user bits.

There are two problems with PDNP. First, when the number of tracks N_t simultaneously processed is greater than one, e.g. in two-dimensional magnetic recording (TDMR), the number of trellis states can become impractically large; this is the *state explosion problem*. Second, the autoregressive noise model and linear prediction used in PDNP is somewhat restrictive and may not accurately represent the media noise, especially at high storage densities; this is the *modeling problem*.

To avoid state explosion, we separate the ISI detection and media noise estimation into two separate detectors and use the turbo-principle to exchange information between them, thus avoiding use of a super-trellis. To address the modeling problem, we design and train DNN-based media noise estimators. DNNs feature a relatively large number (typically ≥ 5) of interconnected network layers, with each network layer containing a hidden layer and non-linear output layer. Recent breakthroughs in DNNs [5], [6] have led to great success in applications such as speech recognition, image understanding, and language translation. DNNs can *learn a probabilistic model from the data.* As DNN models are much more general than autoregressive models, they give a better estimate of magnetic media noise $\mathbf{n}_m$ than PDNP, and hence lead to reduced detector BERs compared to PDNP.

Two previous papers [7], [8] employ neural networks (NNs) for equalization of TDMR channels. However, these NNs have only three layers, and are thus not DNNs, but are more like the first generation NNs introduced in the 1980s. Also, these papers employ the NNs in place of the BCJR/PDNP equalizers, not as media noise predictors, and their performance is compared only with that of a 2D linear equalizer, which is known to be inferior to the 2D-BCJR or Viterbi equalizers typically employed in TDMR detectors (e.g. [9]–[12]).

To our knowledge, the present paper is one of the first to employ a DNN media noise predictor, and to combine it with trellis-based ISI detection in a turbo architecture.

Manuscript received September 3, 2019. Corresponding author: B. Belzer (email: belzer@wsu.edu)

II. System Model

The BCJR-DNN turbo detector assume
for the kth linear equalizer filter output $y(k$
the 1D PDNP scheme described in the int

$$y(k) = (\mathbf{h} * \mathbf{u})(k) + n_m(k) +$$

where $\mathbf{h}$ is the PR target, $\mathbf{u}$ are the code
* indicates 1D convolution, $n_m(k)$ is me
reader electronics AWGN, and the ISI c
$\text{length}(\mathbf{h}) - 1$. Unlike PDNP, the media n
not modeled as an AR process; instead a r
for $n_m(k)$ is learned by the DNN through

We use GFP model data to train and e
The GFP waveforms are generated based
simulations [13]. The simulated media h
11.4 Teragrains per square inch. The GFP waveforms correspond to five tracks of coded bits (± 1), denoted as tracks 0 through 4. They are written using shingled writing technology. Track 0 at the bottom is written first. Then track 1 is written, overlapping part of track 0. The writing process repeats until track 4 is written. Track 4 is called the fat track, since it is not followed by any more tracks and thus preserves the original magnetic write width (MWW), which is 75 nm. In our GFP simulations, the bit length (BL) is 11 nm. We have two GFP data sets for system evaluation. For the first data set the track pitch (TP) (i.e., the distance between adjacent tracks) is 48 nm, and for the second the TP is 27 nm. The number of grains per coded bit (GPB) for the 48 nm TP data set is

$$\text{GPB} = \text{Grain density} \times \text{BL} \times \text{TP} = 9.33.$$

Similarly, for the 27 nm TP data set we compute GPB = 5.25.

Each track in GFP data set consists of 41206 coded bits, close to the sector size of 32768 bits (4K bytes) in a typical HDD. The readings from the center of Track #2 are used as input to the BCJR-DNN turbo detector, and to a comparison baseline 1D PDNP detector.

III. BCJR-DNN Detector

Fig. 1 shows the system block diagram for the proposed BCJR-DNN turbo detector. This system is a turbo-equalization structure that separates the ISI detection and media-noise prediction functions into two detectors that iteratively exchange LLR estimates of coded bits and noise samples until convergence to a low BER occurs.

In Fig. 1, the GFP simulated HDD read-head output vector $\mathbf{r}$ contains two samples per coded bit, denoted $\mathbf{r}^{(1)}$ and $\mathbf{r}^{(2)}$. These samples are on the same track and are collected by the same read head, but are located at different downtrack locations within a given bit; the odd samples $\mathbf{r}^{(1)}$ (the "first samples" per bit) are located near the center of each bit, and the even samples $\mathbf{r}^{(2)}$ are located at the boundary between bits. The odd samples $\mathbf{r}^{(1)}$ are first filtered by a length 15 1D linear equalizer designed to minimize the mean squared error (MMSE) between the filter output $\mathbf{y}^{(1)}$ and the convolution of the coded bits $\mathbf{u}$ with the 1D PR mask $\mathbf{h}$. This PR equalization is done because the down track ISI can have a span of up to about 15 bits. The filter output $\mathbf{y}^{(1)}$ is input to the BCJR

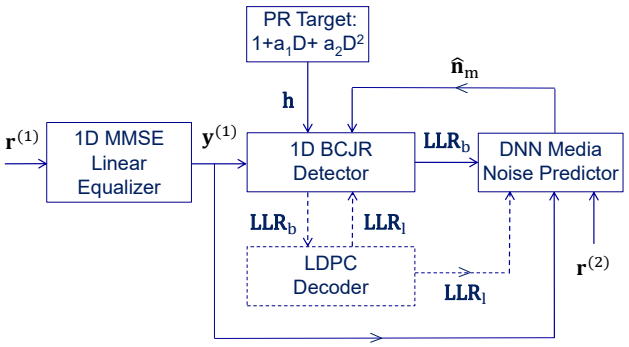


Fig. 1: Block diagram for the BCJR-DNN turbo detector.

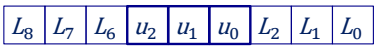
detector, which handles only ISI equalization based on the PR target $\mathbf{h}$ and outputs a block $\mathbf{LLR}_b$ of 41206 coded bit LLRs. In this work, we design the PR target $\mathbf{h}$ with three taps, so that the ISI channel length $I = 2$, and the BCJR detector has $M = 2^I = 4$ states and eight total branches.

The BCJR's coded bit LLRs $\mathbf{LLR}_b$ are sent to the DNN, which provides an estimate $\hat{\mathbf{n}}_m$ of the media noise to the next iteration of the BCJR detector in order to improve the BCJR's estimate $\mathbf{LLR}_b$. In the BCJR's gamma probability computation for the jth trellis branch at trellis stage k, the exponent in the Gaussian conditional channel probability density function (PDF) is $(y_k^{(1)} - (\mathbf{h} * \mathbf{v}_j)_l - \hat{n}_{m_k})^2$, where $\mathbf{v}_j$ denotes the vector branch label for branch j (which has the same length as $\mathbf{h}$), and l is the index corresponding to $y_k^{(1)}$; thus, an accurate estimate of the media noise maximizes this PDF when $\mathbf{v}_j$ corresponds to the correct data bits. The DNN inputs also include the filtered first sample sequence $\mathbf{y}^{(1)}$ and the even reading samples $\mathbf{r}^{(2)}$. We have found experimentally that providing $\mathbf{r}^{(2)}$ as an additional input to the DNN measurably improves the DNN's estimation of the media noise and hence reduces the BER of the BCJR's output in the next iteration.

The dotted lines and box in Fig. 1 indicate future work beyond the present paper; they show how the detector can interface to an LDPC channel decoder.

The BCJR-DNN turbo detection architecture shown in Fig. 1 can be readily generalized to multi-track detection for TDMR. When N_t tracks are simultaneously detected and a 2D linear equalizer and 2D PR target with N_t rows is employed, the BCJR jointly equalizes downtrack ISI as well as inter-track interference (ITI), and its number of trellis states M becomes $M = 2^{N_t I}$; e.g., $M = 64$ when $N_t = 3$ and $I = 2$. *Thus, by limiting the BCJR detector to ISI or ISI-ITI detection only, the trellis state explosion problem is circumvented.*

We employ two methods for interfacing the BCJR detector to the DNN. In the first method, labeled as "1 DNN" in the tabular results provided in section IV, one DNN estimates the media noise for the kth BCJR trellis stage based on $\mathbf{LLR}_b$ (and on $\mathbf{y}^{(1)}$ and $\mathbf{r}^{(2)}$) and then passes this estimate $\hat{n}_{m_k}$ to all eight BCJR branches. In the second method, labeled as "8 DNNs" in section IV, a media noise estimate $\hat{n}_{m_{kj}}$, $0 \le j \le 7$, for the jth branch of the kth BCJR trellis stage is provided by a DNN, denoted DNN_j, dedicated to (and trained for) the jth branch. Thus, at each trellis stage eight separate DNNs (which



(b)

Fig. 2: (a) LLR/state block for 1 DNN: LLRs are from BCJR. (b) LLR/state block for 8 DNNs: branches are specified by fixed value of vector $[u_0, u_1, u_2]$. L_0-L_2 and L_6-L_8 are LLRs from BCJR.

can operate in parallel) provide eight separate media noise estimates, one for each trellis branch. In this second method, the $\mathbf{LLR}_{b_j}$ vector provided to DNN_j has its central three elements equal to the level shifted (to ± 1) three bit binary representation $[u_0, u_1, u_2]$ of the branch index j, and the other elements are unchanged from those in $\mathbf{LLR}_b$. Fig. 2 shows the LLR input vectors for both the 1 DNN and 8 DNNs methods. For this second method, in general, BCJR trellises with N_{br} branches per stage require N_{br} separate DNNs.

A. Deep neural network (DNN)

We investigate two neural network architectures for the noise predictor. The first architecture is the traditional fully connected deep neural network (FCDNN). Each FCDNN layer consists of a hidden layer and an output layer. Each output layer node is connected to a node in the next FCDNN layer's hidden layer through a non-linear activation function. The output o_j of a given output layer's node j is computed from its hidden layer node inputs x_i as $o_j = f(z_j)$, where $z_j = \sum_i w_{ij} x_i + b_j$, and the w_{ij} and b_j are trainable weights and a trainable bias term, respectively. The function $f(z)$ is the rectified linear unit (ReLU) function $f(z) = \max(0, z)$.

The second architecture is the convolutional neural network (CNN), wherein each convolutional layer has a bank of trained finite length filters connected to an output layer. Each output layer node has a trainable bias term, and is connected to the next convolutional layer through a ReLU function. Both the FCDNN and the CNN can perform feature extraction to facilitate media noise prediction.

The DNNs process their input data in a sliding block manner. To estimate the kth media noise sample $\hat{n}_{m_k}$, the lowest input layer of each DNN accepts a block $\mathbf{LLR}_{b_k}$ of N_i BCJR output LLRs, N_i filtered readings $\mathbf{y}_k^{(1)}$, and N_i raw second readings $\mathbf{r}_k^{(2)}$ 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 paper, $N_i = 9$. To estimate the $(k+1)$th media noise sample, each of the input data blocks is shifted by exactly one sample into the future.

1) Fully connected deep neural network (FCDNN)

Fig. 3 shows the architecture for the proposed FCDNN, which we designed for the noise predictor. A FCDNN is a modular layer design for which we can choose the network depth to optimize the model. Every node in each fully connected layer is connected to all the nodes in the previous layer.

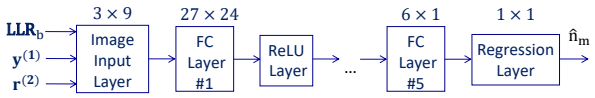


Fig. 3: Fully connected neural network architecture.

The proposed FCDNN has 11 layers. The first layer is an input image layer in which data normalized to have zero mean and unit variance enters. The input layer size is 27 when there are three length-9 input data blocks (i.e., $\mathbf{LLR}_{b_k}$, $\mathbf{y}_k^{(1)}$ and $\mathbf{r}_k^{(2)}$); the input layer size is 18 when only $\mathbf{LLR}_{b_k}$ and $\mathbf{y}_k^{(1)}$ are used as inputs. The output of this layer feeds the next layer, which is a fully connected one. The proposed network has 5 fully connected layers. For three input data blocks, the sizes of fully connected layers 1 through 5 are equal to $[27 \times 24]$, $[24 \times 18]$, $[18 \times 12]$, $[12 \times 6]$, and $[6 \times 1]$; for two input blocks, the sizes are $[18 \times 15]$, $[15 \times 12]$, $[12 \times 9]$, $[9 \times 6]$, and $[6 \times 1]$. After each fully connected layer except the last one, there exists a Rectified Linear Unit (ReLU) activation function layer. The ReLU function assists the model to converge with greater acceleration. After the last fully connected layer, there should not be any ReLU layer since thresholding the last layer's output will give a poor estimate of the true media noise, which is a real number that can take positive or negative values. The last layer is the regression layer, which predicts the responses of the model. The regression loss function is $0.5\times$ the mean squared error between the training label media noise and the DNN prediction of the media noise.

2) Convolutional neural network (CNN)

Fig. 4 presents the CNN architecture. The CNN is similar to the FCDNN in that every node receives some input and predicts the output with non-linearity.

The proposed CNN contains 18 layers. These layers are categorized as one input image layer, 5 convolutional units, 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. The 2D input image layer is of size 27, and includes three rows consisting of nine samples from each of the three input blocks $\mathbf{LLR}_{b_k}$, $\mathbf{y}_k^{(1)}$ and $\mathbf{r}_k^{(2)}$. We also consider the case where $\mathbf{r}_k^{(2)}$ is not included as an input; in this case the input layer is of size 18 and includes two rows of nine samples each from $\mathbf{LLR}_{b_k}$ and $\mathbf{y}_k^{(1)}$. Organizing the 1D input blocks into a 2D array in this manner induces 2D spatial correlation between the blocks. We exploit this spatial correlation by employing trained 2D convolutional filters on all CNN layers.

Every convolutional unit includes 3 layers: convolutional layer, batch normalization layer, and ReLU 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 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. In CNNs designed for three rows of input, all convolutional layers employ filters of size $[3 \times 3]$; for two-row-input CNNs

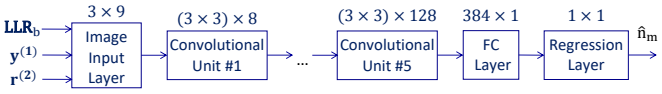


Fig. 4: Convolutional neural network architecture.

the filters are of size $[2 \times 3]$. The number of channels in units 1 through 5 is equal to 8, 16, 32, 64, and 128 respectively.

IV. Simulation Results

In this section, we present simulation results for BCJR-DNN turbo detector on two different GFP waveform data sets. Results are presented for both FCDNN and CNN architectures for the media noise estimator. Simulation results for four scenarios for the DNN input features are also provided.

In all presented BCJR-DNN simulation results, one turbo-loop between the BCJR and the DNN is performed. In this turbo-loop, the BCJR initially assumes that the media noise is zero, and computes an initial set of output LLRs $\mathbf{LLR}_{b_k}$ which are passed to the DNN. The DNN then computes media noise estimates $\hat{n}_{m_k}$ and passes them back to the BCJR. Finally the BCJR is run a second time using the DNN's media noise estimates to obtain a lower BER. The LLR outputs of the second pass through the BCJR are thresholded and used to compute the detector's BER.

A. Data sets

We use two GFP waveform data sets, both with 11 nm BL. The first set has TP = 48 nm and GPB = 9.33; the second has TP = 27 nm and GPB = 5.25. Each block in each data set has $5 \times N_b$ input bits, where $N_b = 41206$. The central three tracks for each data set have two readings per bit, i.e. $3 \times 2N_b$ readings per block. The 1D detectors considered in this paper use only the central track in the GFP waveforms for training and testing. We use 16 GFP blocks as the training data set to train the DNNs and the 1D PDNP used as a comparison baseline. Another (distinct set of) 16 blocks is used as the test data set to generate (by simulation) the BER figures presented in Tables I and II below.

B. Input scenarios

We investigate four scenarios for the DNN input features. In the first, the DNN inputs are the signs of the BCJR detector's output LLRs and the sequence $\mathbf{y}_k^{(1)}$. In the second, the LLR signs are replaced with their corresponding probabilities. For all scenarios, we normalize the data by subtracting the mean and dividing by the standard deviation for each feature, so that the DNN gets unbiased data. The LLR probabilities give more information: not only the estimated bit but also the estimation's reliability can be determined from them. Experiments show that using the LLR probabilities as DNN inputs gives better media noise estimates than using the signed LLR values; this may be due to the non-linear scale inherent in the LLRs, as opposed to the linear scale of their associated probabilities.

The rest of the scenarios utilize the second samples $\mathbf{r}_k^{(2)}$ of the raw GFP readings. For the third scenario, we assign the signs of the LLRs, $\mathbf{y}_k^{(1)}$, and $\mathbf{r}_k^{(2)}$ as the DNN input features. The last scenario has the same features as the third, except we use the LLR probabilities rather than their signs.

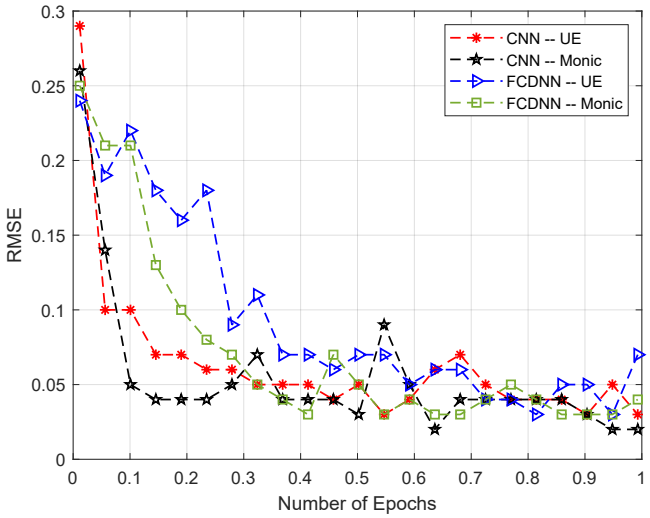


Fig. 5: Learning curves for the FCDNN and CNN on the 48 nm TP data set, when the inputs include the LLR probabilities, $\mathbf{y}_k^{(1)}$, and $\mathbf{r}_k^{(2)}$. One epoch refers to one pass through all 16 training data blocks. The points of the curve indicate separate iterations; an iteration refers to an instance of stochastic-gradient-descent based on a gradient estimate derived from a small subset of the training data.

C. Discussion of simulation results

Figure 5 shows the FCDNN and CNN learning curves (i.e., root MSE (RMSE) versus number of training epochs) for the case when 1 DNN has LLR probabilities, $\mathbf{y}_k^{(1)}$, and $\mathbf{r}_k^{(2)}$ as inputs. Results for three-tap unit energy (UE) and monic PR masks designed using the method described in [14] are included. The convergence speed is fastest, and the final achieved RMSE is lowest, for the CNN with monic mask, followed by the CNN with UE mask, the FCDNN with monic mask, and the FCDNN with UE mask. The first three of these converge within one epoch, whereas the FCDNN with UE mask takes a little longer to converge. The learning curve convergence results are consistent with the BER results presented in Tables I and II below.

Table I summarizes the results for the TP = 48 nm data set. The block of LLR inputs $\mathbf{LLR}_{b_k}$ to the DNN from the BCJR is indicated by $\mathbf{L}$ in Table I. The table compares the BER performance of the proposed BCJR-DNN detector to that of a 1D PDNP BCJR detector with 128 states, corresponding to $I = 2$, $L = 4$, and $\Delta = 1$. The 1D PDNP takes its input from the same length 15 MMSE filter (with the same PR target $\mathbf{h}$) as that used with the BCJR-DNN detector. The PDNP's pattern vector length of $I + 1 + L + \Delta = 8$ bits is about equal to the DNN's channel input $\mathbf{y}^{(1)}$ length of 9 samples.

To test whether the PDNP can exploit both sample sequences $\mathbf{r}_k^{(1)}$ and $\mathbf{r}_k^{(2)}$ without doubling the number of state bits, we compute the averaged input $\mathbf{r}_k^{\text{avg}} = (\mathbf{r}_k^{(1)} + \mathbf{r}_k^{(2)})/2$ and

TABLE I: Simulations of PDNP and DNN/BCJR detectors at TP 48 nm.

Method	Input	PR Mask	BER No AWGN		BER SNR 20 dB	
PDNP	Single Sample	UE	**4.55e−4**		7.54e−4	
		Monic	5.55e−4		**5.76e−4**	
	Double Samples	UE	7.45e−4		1.32e−3	
		Monic	7.68e−4		9.01e−4	
DNN			**8 DNNs**	**1 DNN**	**8 DNNs**	**1 DNN**
FC	Sign[**L**], $\mathbf{y}^{(1)}$	UE	1.10e−3	1.02e−3	3.59e−3	3.38e−3
		Monic	2.84e−4	2.81e−4	7.34e−4	7.24e−4
	Pr[**L**], $\mathbf{y}^{(1)}$	UE	7.58e−4	7.22e−4	2.38e−3	2.42e−3
		Monic	2.29e−4	2.18e−4	5.39e−4	5.51e−4
	Sign[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	2.93e−4	4.08e−4	3.38e−4	3.61e−4
		Monic	2.50e−4	2.79e−4	2.91e−4	3.85e−4
	Pr[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	2.58e−4	2.34e−4	3.29e−4	2.61e−4
		Monic	2.09e−4	2.09e−4	2.64e−4	2.55e−4
CNN	Sign[**L**], $\mathbf{y}^{(1)}$	UE	3.06e−4	3.52e−4	7.72e−4	9.57e−4
		Monic	1.58e−4	1.67e−4	4.14e−4	4.40e−4
	Pr[**L**], $\mathbf{y}^{(1)}$	UE	2.93e−4	2.85e−4	7.77e−4	7.62e−4
		Monic	1.65e−4	1.65e−4	4.41e−4	4.32e−4
	Sign[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	1.87e−4	2.28e−4	2.08e−4	2.43e−4
		Monic	1.53e−4	1.70e−4	**1.97e−4**	2.09e−4
	Pr[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	1.85e−4	1.64e−4	2.25e−4	2.14e−4
		Monic	**1.52e−4**	**1.62e−4**	2.02e−4	**2.00e−4**

TABLE II: Simulations of PDNP and DNN/BCJR detectors at TP 27 nm.

Method	Input	PR Mask	BER No AWGN		BER SNR 20 dB	
PDNP	Single Sample	UE	9.75e−3		1.58e−2	
		Monic	**7.12e−3**		**1.12e−2**	
	Double Samples	UE	1.25e−2		2.35e−2	
		Monic	8.75e−3		1.73e−2	
DNN			**8 DNNs**	**1 DNN**	**8 DNNs**	**1 DNN**
FC	Sign[**L**], $\mathbf{y}^{(1)}$	UE	2.08e−2	2.16e−2	3.22e−2	3.64e−2
		Monic	7.30e−3	7.39e−3	1.18e−2	1.26e−2
	Pr[**L**], $\mathbf{y}^{(1)}$	UE	1.86e−2	1.67e−2	2.83e−2	2.90e−2
		Monic	7.20e−3	7.25e−3	1.19e−2	1.18e−2
	Sign[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	8.52e−3	9.96e−3	1.13e−2	1.24e−2
		Monic	7.08e−3	7.22e−3	9.51e−3	1.03e−2
	Pr[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	8.36e−3	8.44e−3	1.10e−2	1.13e−2
		Monic	7.12e−3	7.14e−3	9.60e−3	9.58e−3
CNN	Sign[**L**], $\mathbf{y}^{(1)}$	UE	1.14e−2	1.21e−2	1.94e−2	2.17e−2
		Monic	6.92e−3	7.08e−3	1.16e−2	1.16e−2
	Pr[**L**], $\mathbf{y}^{(1)}$	UE	1.10e−2	1.14e−2	1.86e−2	1.90e−2
		Monic	6.96e−3	6.99e−3	1.15e−2	1.15e−2
	Sign[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	7.72e−3	8.68e−3	9.97e−3	1.07e−2
		Monic	6.74e−3	7.06e−3	9.18e−3	9.66e−3
	Pr[**L**], $\mathbf{y}^{(1)}$, $\mathbf{r}^{(2)}$	UE	7.74e−3	7.78e−3	9.84e−3	9.87e−3
		Monic	**6.71e−3**	6.83e−3	**8.45e−3**	9.16e−3

design the entire BCJR-PDNP system (including the MMSE filter, the PR mask, and trained PDNP coefficients) for the input $\mathbf{r}_k^{\text{avg}}$; the results for this case are shown in the row labeled "Double Samples" under the PDNP method. The BER for the double sample case is higher in all cases than that of the single sample case that only uses $\mathbf{r}^{(1)}$. This probably occurs because $\mathbf{r}^{(2)}$ is not co-located with $\mathbf{r}^{(1)}$, and thus averaging it with $\mathbf{r}^{(1)}$ does not give true noise averaging. In addition, the location of $\mathbf{r}^{(2)}$ between the bits introduces additional downtrack ISI into the average of the two readings.

The GFP data contains no read-head electronic AWGN, i.e., $n_e(k) = 0$ in (1). The column labeled "BER No AWGN" reports results for this case. The column labeled "BER SNR 20 dB" reports results when non-zero AWGN $n_e(k)$ at an SNR of 20 dB is added to both sample sequences $\mathbf{r}_k^{(1)}$ and $\mathbf{r}_k^{(2)}$. The SNRs for the cases when the PDNP (or DNN) uses only $\mathbf{r}_k^{(1)}$ or both samples $\mathbf{r}_k^{(1)}$ and $\mathbf{r}_k^{(2)}$ are computed as

$$\text{SNR}_1 = 10\log_{10}\left(\frac{1}{\sigma_e^2}E\left[(r_k^{(1)})^2\right]\right) \quad (2)$$

$$\text{SNR}_2 = 10\log_{10}\left(\frac{1}{\sigma_e^2}E\left[(r_k^{(1)})^2+(r_k^{(2)})^2\right]\right), \quad (3)$$

where SNR_1 and SNR_2 indicate the single and double sample cases, and σ_e^2 is the AWGN variance.

For the zero AWGN results, except for a few UE mask cases, the BCJR-DNN detector achieves lower BERs than the PDNP detector. The BCJR-DNN detector's lowest BER of 1.52e-4 occurs in the last row of the table with the monic mask, 8 CNN noise predictors, and inputs of the LLR probabilities Pr[**L**] and $\mathbf{y}_k^{(1)}$ and $\mathbf{r}_k^{(2)}$; this BER is about $0.334\times$ the PDNP's lowest BER of 4.55e-4, which is achieved with the UE mask. The next lowest BER of 1.62e-4 (i.e. $0.356\times$ the PDNP's lowest BER) occurs under the same conditions except with only 1 CNN.

For the BCJR-DNN detector, the monic mask gives lower BER than the UE mask in all cases. By contrast, the UE mask gives the lowest BER for the 1D PDNP detector. Using LLR probabilities in the BCJR-DNN detector always gives lower BERs than using LLR signs. Adding the second GFP sample $\mathbf{r}_k^{(2)}$ to the DNN inputs always lowers the BER, although the reduction is less for the monic mask than for the UE mask.

In many cases there is little to no advantage in using 8 DNNs instead of 1 DNN, although 8 DNNs do give a significant BER reduction with the UE mask, LLR signs and both GFP input samples. Also, 8 CNNs with the monic mask achieve the lowest overall BER in the table's last row.

The BERs for AWGN at 20 dB SNR are higher than those with zero AWGN; the single exception occurs with one UE mask FCDNN system that uses the sign of the LLR and both readings. The non-linear sign operation might actually benefit from some amount of AWGN. Trends with the 20 dB SNR results are similar to those of the zero AWGN case: the lowest CNN BER of 1.97e-4 (essentially tied between 1 CNN and 8 CNNs) is about $0.342\times$ the lowest PDNP BER of 5.76e-4.

Table II summarizes results for the TP = 27 nm data set. There is significantly more ITI, which neither 1D detector can explicitly take into account; this leads to higher BERs throughout Table II as compared with Table I. The trends are similar to those in Table I, except that the zero-AWGN BCJR-DNN detector's BER is lower than the PDNP's BER only for the monic-mask CNN cases in the table's last and second to last rows, and the monic-mask 8-FCDNN case in the table's sixth FCDNN row. The lowest BCJR-DNN BER is about $0.942\times$ the PDNP's lowest BER for zero AWGN, and about $0.754\times$ lower for AWGN at 20 dB SNR. PDNP's superior robustness to unknown ITI may be due to its use of

TABLE III: Computational complexity for BCJR-PDNP, BCJR-1 CNN and BCJR-8 CNNs.

Method	mul/div	add/sub	exp/log	sqrt
PDNP	141,168	41,347	257	256
1 DNN	12,953	11,087	18	16
8 DNNs	101,013	86,528	18	16

256 different prediction filters (conditioned on the super-trellis branch) compared to the BCJR-DNN detector's much lower degree of branch conditioning. This hypothesis is supported by the generally superior performance of 8 DNNs over 1 DNN in Table II, which is not the case in Table I.

We also investigated a second iteration between the BCJR and the noise-prediction CNN. In this scenario, a second-iteration CNN is trained with the BCJR output LLRs that result from the first-iteration CNN's noise prediction (as well as with $\mathbf{y}_k^{(1)}$ and $\mathbf{r}_k^{(2)}$), and the BCJR uses the second-iteration CNN's noise prediction to derive new LLR estimates of the data bits. However, the second iteration did not reduce the final BER. Further investigation of design of the second-iteration CNN is left for future work.

D. Computational run time and complexity comparison

We measured the run time per input bit by running all 16 test blocks on the same CPU for the PDNP, BCJR-FCDNN and BCJR-CNN detectors; both BCJR-DNN detectors use one DNN. The PDNP required 774 μs per bit, the BCJR-CNN required 89.6 μs per bit, and the BCJR-FCDNN required 43.7 μs per bit. Thus, both BCJR-DNN detectors run substantially faster than the PDNP. The best performing BCJR-CNN detector with 8 CNNs requires 0.731 times the per bit running time (PBRT) of the PDNP detector; running all 8 CNNs in parallel would reduce the PBRT by about $1/8$th. The next best performing detector uses only 1 CNN and requires only 0.116 times the PBRT of the PDNP detector.

The run-time comparisons are supported by the computational complexity figures shown in Table III. The complexity of the BCJR algorithm grows as the square of the number of states, due to the double summation over the state variables required to compute the LLRs. The higher complexity of the BCJR-PDNP is due to its trellis having $32\times$ the number of states of the trellis used in the BCJR-CNN system.

V. Conclusion

This paper has presented a concatenated BCJR-DNN architecture for detection of bits on single-track HDDs. Separation of the ISI detection and media noise estimation between the BCJR and DNN detectors avoids the high complexity associated with use of super-trellises in PDNP detectors. The DNN's ability to learn general probabilistic models from the data enables the BCJR-DNN detector to achieve lower BERs than the PDNP detector. In tests with GFP waveforms at 48 nm track pitch and 11 nm bit length, a BCJR-CNN detector with 8 DNNs achieves $0.334\times$ the BER and $0.731\times$ the PBRT of a 1D PDNP detector; a version with 1 DNN achieves $0.356\times$ the BER and $0.116\times$ the PBRT of the PDNP detector. The BCJR-DNN detector can be readily generalized for detection of TDMR waveforms, and can be interfaced with an LDPC code in order to demonstrate what we believe will be substantial areal density gains over either 1D or 2D PDNP; these generalizations will be the topic of future publications.

Acknowledgment

This work was supported by NSF grant CCF-1817083.

References

[1] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*. New York: McGraw-Hill, 1979.

[2] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284–287, Mar. 1974.

[3] A. Kavcic and J. M. Moura, "The Viterbi algorithm and Markov noise memory," *IEEE Trans. Inform. 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 Jour. Sel. Areas Commun.*, vol. 19, no. 4, pp. 730–743, Apr 2001.

[5] G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comp.*, vol. 18, pp. 1527–1554, 2006. [Online]. Available: http://www.cs.toronto.edu/~fritz/absps/ncfast.pdf

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

[7] M. Yamashita, H. Osawa, Y. Okamoto, Y. Nakamura, Y. Suzuki, K. Miura, and H. Muraoka, "Read/write channel modeling and two-dimensional neural network equalization for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 47, pp. 3558–3561, Oct 2011.

[8] M. Yamashita, Y. Okamoto, Y. Nakamura, H. Osawa, K. Miura, S. J. Greaves, H. Aoi, Y. Kanai, and H. Muraoka, "Modeling of writing process for two-dimensional magnetic recording and performance evaluation of two-dimensional neural network equalizer," *IEEE Trans. Magn.*, vol. 48, pp. 4586–4589, Nov 2012.

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

[10] 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, pp. 1–7, July 2016.

[11] ——, "Micromagnetics-based analysis of multi-track detection with simplified 2-D write precompensation on shingled magnetic recording," *IEEE Trans. Magn.*, vol. 52, pp. 1–11, Sept 2016.

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

[13] K. S. Chan, R. Radhakrishnan, K. Eason, M. R. Elidrissi, J. J. Miles, B. Vasic, and A. R. Krishnan, "Channel models and detectors for two-dimensional magnetic recording," *IEEE Trans. Magn.*, vol. 46, no. 3, pp. 804–811, March 2010.

[14] 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, pp. 1–15, Oct. 2015, article no. 3101215.