Deep Transfer Learning-Based Detection for Flash Memory Channels

Zhen Mei, *Member, IEEE*, Kui Cai, *Senior Member, IEEE*, Long Shi, *Senior Member, IEEE*, Jun Li, *Senior Member, IEEE*, Li Chen, *Senior Member, IEEE*, and Kees A. Schouhamer Immink, *Life Fellow, IEEE*

***Abstract*— The NAND flash memory channel is corrupted by different types of noises, such as the data retention noise and the wear-out noise, which lead to unknown channel offset and make the flash memory channel non-stationary. In the literature, machine learning-based methods have been proposed for data detection for flash memory channels. However, these methods require a large number of training samples and labels to achieve a satisfactory performance, which is costly. Furthermore, with a large unknown channel offset, it may be impossible to obtain enough correct labels. In this paper, we reformulate the data detection for the flash memory channel as a transfer learning (TL) problem. We then propose a model-based deep TL (DTL) algorithm for flash memory channel detection. It can effectively reduce the training data size from 10^6 samples to less than 10^4 samples. Moreover, we propose an unsupervised domain adaptation (UDA)-based DTL algorithm using moment alignment, which can detect data without any labels. Hence, it is suitable for scenarios where the decoding of error-correcting code fails and no labels can be obtained. Finally, a UDA-based threshold detector is proposed to eliminate the need for a neural network. Both the channel raw error rate analysis and simulation results demonstrate that the proposed DTL-based detection schemes can achieve near-optimal bit error rate (BER) performance with much less training data and/or without using any labels.**

***Index Terms*— Data detection, error correction code, flash memory, neural network, transfer learning.**

I. Introduction

AS A TYPE of emerging non-volatile memories (NVMs), NAND flash memory has been widely applied in various storage systems, ranging from mobile devices to data centers. To increase the storage density, flash memory technologies have been evolved from one bit per cell (single-level-cell (SLC)) to a maximum of four bits per cell (quad-level-cell (QLC)) [1]. However, the raw bit error rate (RBER) of flash memories with multiple bits per cell becomes larger due to various impairments of the system that are difficult to be predicted and compensated [1] before the decoding of error correction codes (ECCs). In particular, the wear-out noise caused by program/erase (P/E) cycling and the data retention noise caused by charge leakage over time dominate the source of errors, and they lead to unknown channel/cell threshold voltage offset and make the flash memory channel non-stationary [2].

In this paper, we consider NAND flash memory with q bits stored in each memory cell, which results in 2^q possible cell states. The threshold voltage of each state can be represented by a probability density function (PDF). As an example, the threshold voltage distributions for multi-level cell (MLC) ($q = 2$) NAND flash memory are illustrated in Fig. 1. The boundaries V_a, V_b, V_c between neighboring states are referred as read reference voltages or read thresholds, which are used to differentiate the states upon reading the memory cell. When the memory cells are corrupted by various noises, the PDF of these states changes, making the original read thresholds no longer optimal. These sub-optimal read thresholds will lead to more raw bit errors, which severely affect the reliability of the flash memory [1].

To mitigate such performance degradation of flash memories, ECCs have been employed to correct a certain amount of errors. To correct the multiple-bit errors, Bose-Chaudhuri-Hocquenghem (BCH) codes have been widely applied in practical flash memories [3]. As the technology scales down, more powerful ECCs such as the low-density parity-check (LDPC) codes have been proposed to provide higher error-correction capability [4], [5]. Since ECCs correct the errors with the cost of sacrificing data storage efficiency, better channel/data detection schemes that can effectively reduce the

Manuscript received 16 May 2023; revised 30 September 2023 and 20 December 2023; accepted 14 January 2024. Date of publication 23 January 2024; date of current version 18 June 2024. This work was supported by the National Natural Science Foundation of China under Grants 62201258 and 62071498, by the open research fund of National Mobile Communications Research Laboratory, Southeast University (No. 2023D12), by "the Fundamental Research Funds for the Central Universities", No.30923011035, by the Jiangsu Specially-Appointed Professor Program 2021, by the Singapore Ministry of Education Academic Research Fund Tier 2 T2EP50221-0036 and RIE2020 Advanced Manufacturing and Engineering (AME) programmatic grant A18A6b0057. An earlier version of this paper was presented in part at the IEEE International Conference on Communications (ICC), May 2023 [DOI: 10.1109/ICC45041.2023.10279375]. The associate editor coordinating the review of this article and approving it for publication was L. Schmalen. *(Corresponding authors: Kui Cai; Jun Li.)*

Zhen Mei is with the School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China, and also with the National Mobile Communications Research Laboratory, Southeast University, Nanjing 210096, China (e-mail: meizhen@njust.edu.cn).

Kui Cai is with the Science, Mathematics and Technology Cluster, Singapore University of Technology and Design, Singapore 487372 (e-mail: cai_kui@sutd.edu.sg).

Long Shi and Jun Li are with the School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: slong1007@gmail.com; jun.li@njust.edu.cn).

Li Chen is with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou 510006, China (e-mail: chenli55@mail.sysu.edu.cn).

Kees A. Schouhamer Immink is with Turing Machines Inc., 3016 DK Rotterdam, The Netherlands (e-mail: immink@turing-machines.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCOMM.2024.3357616.

Digital Object Identifier 10.1109/TCOMM.2024.3357616

0090-6778 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

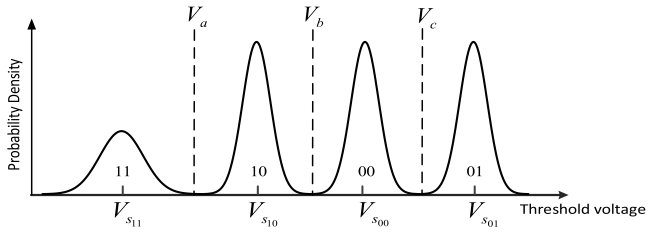


Fig. 1. The initial threshold voltage distributions of MLC ($q = 2$) NAND flash memory.

channel RBER before ECC decoding are needed. As illustrated in the literature, the most effective approach is to adjust the read thresholds with the change of channel conditions to achieve the best RBER performance, which is also the main focus of this paper.

A. Related Works

The design of read thresholds for the flash memories has been investigated in many literature works [6], [7], [8], [9], [10]. They can be classified into two types: the model-driven method and the data-driven method. The model-driven method assumes that the flash memory channel can be modelled by given distributions of the cell threshold voltages, and it then designs the read thresholds based on the estimated PDFs. Specifically, the work of [11] proposed to model the MLC flash memory channel by Gaussian mixture PDF, and then estimated the distribution parameters by gradient descent and Levenberg-Marquardt methods. To better capture the noise statistics, different distributions such as the Beta distribution, Log-normal distribution and Student's t-distribution were also proposed to model the noise PDF [12], [13] of the flash memory channel. For given noise distributions, the read thresholds or quantization boundaries can be designed by maximizing the mutual information (MMI) of the channel [7], or by optimizing some other information theoretic criteria [8], [9]. However, all these model-based methods assume that the channel modeling and the online parameter estimation are accurate, which is difficult to achieve due to the complication of memory physics for various noises.

On the other hand, the data-driven method does not need to know the *a priori* noise PDF. One typical approach is to try different read thresholds until the ECC successfully decodes the codeword [14]. However, this read-retry scheme requires many times of read operations, which lead to a large latency and power consumption. Furthermore, it cannot guarantee to find the optimal read thresholds. Recently, the machine learning-based methods were also proposed to optimize the read thresholds and log-likelihood ratios (LLRs) for ECC decoding [10], [15], [16], [17]. In particular, a deep learning (DL)-based framework to design read thresholds was first proposed in [10], and both recurrent neural network (RNN) and convolutional neural network (CNN)-based detectors were shown to achieve the near-optimal BER performance [10], [15]. In [16], an unsupervised expectation maximization (EM) algorithm was proposed to estimate the channel transition probabilities of the flash memory channel. However, only the binary asymmetric channel (BAC) was actually considered in [16], and the proposed EM algorithm needs to be iterated with the ECC decoder, leading to a very large latency. In [17], a machine learning-based LLR estimation scheme was proposed for flash memories. Although it can directly estimate the LLR, it needs to be invoked for each data block and thereby also results in a large latency and power consumption. Moreover, it is difficult to obtain the optimal LLR using training labels in the presence of the unknown channel offset/noise.

B. Motivations and Contributions of This Work

Although the DL-based read thresholds design can approach the optimal BER performance of the MLC flash memory channel [10], [15], as will be shown in Section III-B, it needs to have a large amount of training samples and labels to achieve a satisfactory performance, which is costly in practice. Even if the neural network can be activated periodically to update the read thresholds, the channel mismatch may still lead to significant performance degradation. Moreover, when the channel offset is large (e.g. caused by a long retention time or a large number of P/E cycles), it will be difficult to obtain enough labels for training. Therefore, it is highly desired to design a DL-based detection approach that can be adapted to unknown channel offset with only a small amount of training samples.

To accomplish this objective, it is necessary to leverage the characteristics of the flash memory channel. It is known that during the early stages of flash memory's lifespan, the channel condition is good and sufficient training samples and labels can be readily acquired. However, as the number of P/E cycles and retention time increase, the noises will severely degrade the performance, and it will become increasingly challenging to obtain labels through ECC decoding. Hence, it is desired to leverage the information from the early stages to reduce the learning difficulty at the targeted P/E cycles and retention time. Inspired by human's capability of learning knowledge from past experience, transfer learning (TL) was proposed to exploit the knowledge and experience gained from a related task to improve the performance of the target task [18], which is highly suitable to our application.

There are different types of TL approaches reported in the literature, such as the instance-based TL, the feature-based TL, and the parameter-based TL [19], [20], [21], [22], [23], [24]. Recently, with the emerge of deep learning techniques, the deep TL (DTL) has also been proposed by integrating TL with deep neural networks [25], [26]. With TL or DTL, the knowledge from the source domain can be transferred to the target domain under the condition that a connection/similarity exists between the two domains. This will significantly alleviate the training difficulty in the target domain. In particular, as a type of TL, the unsupervised domain adaptation (UDA) techniques such as maximum mean discrepancy (MMD) [27] and correlation alignment (CORAL) [21] were proposed to transfer knowledge from a labeled source domain to an unlabeled target domain [28].

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

Inspired by the above ideas, in this work, we propose a DTL framework for the data detection of flash memory channels to reduce the required training samples and labels. Our contributions are summarized as follows.

1) We formulate the data detection of the flash memory channel as a TL problem. We propose a model-based DTL algorithm which can reduce the number of required training samples and labels by two orders of magnitude.
2) To cope with the situations where labels are difficult to obtain, we propose a UDA-based DTL algorithm by aligning the first-order moments of the source domain and target domain, based on which the neural network can be trained without any labels in the target domain.
3) Inspired by the UDA-based DTL algorithm, we further propose a simple UDA-based threshold detector such that the neural network is not required in both the source and the target domains.
4) We also derive the symbol error rate (SER) and RBER for the uncoded MLC and triple-level-cell (TLC) flash memory channels as the performance benchmark.

Our proposed DTL framework can not only be used for the data detection of the flash memory channels, but also be applied to other data storage or communication channels with the non-stationary nature. The proposed DTL algorithms can be directly applied to data detection with quantized signals. Furthermore, although in this work we adopt the RNN for data detection, the model-based DTL and UDA-based DTL algorithms can be applied to other types of neural networks as well.

The rest of this paper is organized as follows. The basics of NAND flash memories and the corresponding channel model are introduced in Section II. In Section III, we present the RNN-based data detection scheme for flash memories, and the effect of training data size is investigated. In Section IV, we formulate the data detection for flash memories as a TL problem, and we propose two DTL algorithms and a UDA-based threshold detector to effectively reduce the required training samples and labels. Experiment results are illustrated in Section V. Finally, Section VI concludes the paper.

II. Preliminaries

A. NAND Flash Memory Basics

In NAND flash memory, q-bit data is stored as the threshold voltage in each flash memory cell. The possible states of memory cells are denoted as $\{s_0, s_1, \ldots, s_{2^q-1}\}$, where s_0 is known as the erased state, and other states are programmed states. For example, for a MLC flash memory, there are four possible states $\{s_0, s_1, s_2, s_3\}$, and a Gray mapping can be used to represent the bit mapping of each state given by $\{11, 10, 00, 01\}$. Similarly, TLC flash memory has eight possible states $\{s_0, s_1, \ldots, s_7\}$ and the corresponding bit mapping can be taken as $\{111, 110, 100, 000, 010, 011, 001, 101\}$.

However, the threshold voltage of each state will shift and their distributions overlap due to various types of noises in flash memories. This will result in decision errors and serious degrade the data recovery performance. There are four major sources of errors, namely, programming noise, data retention noise, wear-out noise, and the cell-to-cell interference (CCI) [6], [29]. The characteristics of these four types of noises are described as follows:

1) Programming Noise: Each flash memory cell is a floating gate and its threshold voltage can be configured by transferring charges into the floating gate. However, process variations will lead to the programming noise n_p of each voltage state, which follows a Gaussian distribution with zero mean and variance of σ_e^2 or σ_p^2 [29], [30], [31]. Here, σ_e^2 denotes the noise variance of the erased state voltage v_{s_0}, and σ_p^2 represents that of each programmed state voltage $\{v_{s_1}, v_{s_2}, \ldots, v_{s_{2^q-1}}\}$. Therefore, we have

$$p_{n_p}(v) = \begin{cases} \mathcal{N}(0, \sigma_e^2), & \text{for } v \in \{v_{s_0}\} \\ \mathcal{N}(0, \sigma_p^2), & \text{for } v \in \{v_{s_1 \sim s_{2^q-1}}\} \end{cases}. \quad (1)$$

Then, this programming procedure is performed by repeatedly pulsing the voltage with a step voltage $\triangle V_{pp}$, which is known as incremental-step-pulse programming (ISPP) [32]. The ISPP noise n_i only affects the programmed states $\{v_{s_1}, v_{s_2}, \ldots, v_{s_{2^q-1}}\}$, while the erase state voltage v_{s_0} is not affected by n_i. It causes the voltage of the programmed state memory cells to follow a uniform distribution [6], given by

$$p_{n_i}(v) = \begin{cases} \dfrac{1}{\triangle V_{pp}}, & V_p \le v \le V_p + \triangle V_{pp} \\ 0, & \text{otherwise,} \end{cases}, \quad (2)$$

where $V_p \in \{V_{s_{10}}, V_{s_{00}}, V_{s_{01}}\}$. Hence, the overall distribution of the erase state is $p_{n_p}(v)$, and that of programmed states is the convolution of $p_{n_p}(v)$ and $p_{n_i}(v)$.

2) Data Retention Noise: The data retention noise n_r is caused by charge leakage after the memory cell is being programmed, and it results in a threshold voltage drop over time. Specifically, as the retention time increases, the threshold voltage v_p will shift towards that of the lower-voltage states, while the erased state voltage v_{s_0} is almost unchanged. Moreover, the voltage shift of higher-voltage states is larger than that of the lower-voltage states. Following [30], [33], [34], the data retention noise can be model by a Gaussian distribution with mean μ_{r_s} and standard deviation σ_{r_s}, given by [29]

$$\mu_{r_s} = (V_s - x_0) \cdot (A_t N_{\text{PE}}^{\alpha_i} + B_t N_{\text{PE}}^{\alpha_o}) \cdot \ln(1 + T), \quad (3)$$

$$\sigma_{r_s} = 0.3|\mu_{r_s}|, \quad (4)$$

where V_s is the desired write voltage level with $s \in \{s_0, s_1, \ldots, s_{2^q-1}\}$, N_{PE} denotes the number of P/E cycles, T is the retention time, and $(x_0, A_t, B_t, \alpha_i, \alpha_o)$ are constants.

3) Wear-Out Noise: The wear-out noise n_w is caused by the repeated P/E cyclings that damage the oxide layer of floating gate transistors [35]. The wear-out noise tends to widen the threshold voltage distributions and can be modelled by a Gaussian or exponential distribution [30], [36] with zero mean and standard deviation of $\sigma_w = 0.00027\ N_{\text{PE}}^{0.62}$ [29].

4) CCI: Apart from the above noises, the threshold voltage of a flash memory cell may also be affected by the programming of its adjacent cells [32], which is known as the CCI. The CCI happens due to parasitic capacitance coupling between memory cells. The interference from the

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

adjacent cells is linearly added to the threshold voltage of a victim cell. As reported in the literature [37], the CCI can be effectively mitigated by pre-distortion or post-processing techniques. Hence, in this work, we assume that the CCI has already been removed.

B. Channel Model

The overall threshold voltage distributions of flash memory cells can be computed as the convolution integral of all the noise components, and it can be well approximated by the Gaussian distribution [36]. In this paper, we first adopt the Gaussian model to generate noise samples for simulations. The combined noises can be expressed as

$$v = V_s + n_i + n_p + n_r + n_w. \tag{5}$$

Each noise component in (5) follows the Gaussian distribution and the final combined means and variances are given by

$$\mu_{s_0} = V_{s_0} - \mu_{r_{s_0}}, \tag{6}$$

$$\mu_{s_1 \sim s_{2^q-1}} = V_{s_1 \sim s_{2^q-1}} + \frac{\triangle V_{pp}}{2} - \mu_{r_{s_1 \sim s_{2^q-1}}}, \tag{7}$$

$$\sigma^2_{s_0} = \sigma^2_e + \sigma^2_w + \sigma^2_{r_{s_0}}, \tag{8}$$

$$\sigma^2_{s_1 \sim s_{2^q-1}} = \sigma^2_p + \sigma^2_w + \sigma^2_{r_{s_1 \sim s_{2^q-1}}}, \tag{9}$$

for the erased-state cell and programmed-state cell, respectively. In the simulations, we adopt the parameters from [29] and assume $\triangle V_{pp} = 0.2$, $\sigma_e = 0.35$, $\sigma_p = 0.05$, $x_0 = 1.4$, $A_t = 0.000035$, $B_t = 0.000235$, $\alpha_i = 0.62$, and $\alpha_o = 0.3$.

Note that non-Gaussian distributions such as the Beta distribution and the Gamma distribution can be used to accurately model the heavy-tail of realistic threshold voltage distributions [12]. To further verify our proposed DTL-based detection approaches in more realistic scenarios, the Gamma distribution is also employed in our simulations.

The above channel model is a simplified model of the flash memory channels that is widely adopted by many literature works [11], [12], [36]. Due to the lack of experimental data from the practical flash memories, this channel model is only used to generate data for training and testing the NNs. Note that our transfer learning approaches are not restricted to a specific channel model, and the proposed DTL-based detectors are data-driven and do not require any knowledge of the channel.

III. Neural Network-Based Data Detection Without TL

A. RNN-Aided (RNNA) Threshold Detection

Similar to [10], the data detection of the flash memory channels is formulated as a machine learning problem and the NN is employed to accomplish the task. This problem can be seen as either a classification task or a regression task. In this work, to efficiently detect multiple data symbols for each NN inference, the data detection is regarded as a regression task. Specifically, the readback threshold voltage of the k-th memory cell is denoted by v_k. The input of the NN is given by $\boldsymbol{v} = \{v_1, v_2, \cdots, v_N\}$, where N is the input size of the NN. The outputs of the NN are the estimates $\tilde{\boldsymbol{x}} = \{\tilde{x}_1, \tilde{x}_2, \cdots, \tilde{x}_N\}$

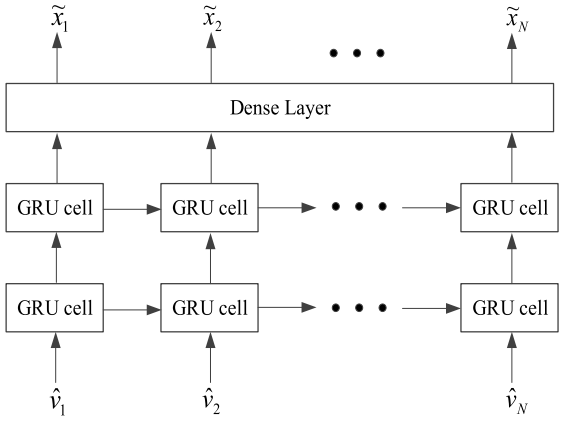


Fig. 2. The stacked RNN architecture for data detection.

of the labels $\boldsymbol{x}$. To conveniently derive the read thresholds and demap labels to binary bits for ECC decoding subsequently, the 2^q voltage states $\{v_{s_0}, \ldots, v_{s_{2^q-1}}\}$ of the flash memory cell are labeled as $\{0, 1, \ldots, 2^q - 1\}$, respectively. Therefore, if we denote the set of network parameters as $\boldsymbol{\theta}$, the neural network output can be expressed as

$$\tilde{\boldsymbol{x}} = f(\boldsymbol{v}, \boldsymbol{\theta}), \tag{10}$$

where $f(\cdot)$ represents the neural network. The stored data in the k-th memory cell can be detected by rounding the neural network output $\tilde{x}_k$ to its nearest integer and then demapping it back to binary bits. Our task is to find an neural network model $f(\cdot)$ and the corresponding parameters $\boldsymbol{\theta}$ such that the detection error probability is minimized.

As illustrated by Fig. 2, we employ the same stacked RNN architecture as proposed in [10]. It consists of two gated recurrent unit (GRU) layers and one fully-connected output layer. For the output layer, an additional softplus activation function is used to introduce non-linearity to the neural network, given by $\sigma_{\text{softplus}}(t) = \ln(1 + \exp(t))$, with $\sigma_{\text{softplus}}(t) \in [0, \infty)$. Once the RNN architecture is determined, we can train the neural network to find model parameters such that the loss function $\mathcal{L}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ is minimized. In this work, we choose the mean square error (MSE) as the loss function, given by

$$\mathcal{L}(\boldsymbol{x}, \tilde{\boldsymbol{x}}) = \frac{1}{N} \sum_{k=1}^{N} (x_k - \tilde{x}_k)^2. \tag{11}$$

By using the gradient descent-based algorithms and back propagation, the optimized $\boldsymbol{\theta}$ can be obtained by minimizing $\mathcal{L}(\boldsymbol{x}, \tilde{\boldsymbol{x}})$ over the entire training data set. After training, we can employ the trained RNN with the optimized $\boldsymbol{\theta}$ to detect the data. The corresponding network settings and parameters are given in Table I.

Similar to [10], we can derive updated read thresholds based on the RNN outputs, leading to the RNN-aided (RNNA) threshold detector. It only needs to be activated periodically when the system is in the idle state. After that, the detection can be carried out directly using the updated thresholds. To obtain labels $\boldsymbol{x}$ for training, we can use codewords that are correctly decoded by the ECC decoder as labels. However, if the channel is severely contaminated by noises, it will be difficult to obtain enough labels for training.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

TABLE I
NETWORK SETTINGS AND HYPER-PARAMETERS

Batch Size	20
Number of Epochs	50
Loss Function	MSE
Initializer	Xavier uniform initializer
Optimizer	Adam optimizer

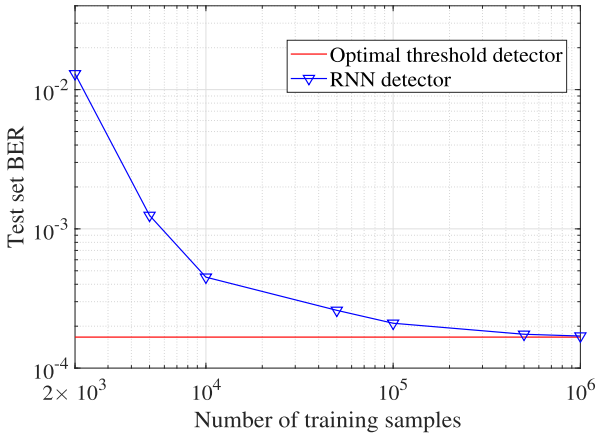


Fig. 3. RBER performance of the optimum threshold detector and the RNN detector with different number of training samples at $N_{\text{PE}} = 10^3$ and $T = 10^3$ hours.

B. The Effects of Training Data Size

As a supervised learning approach, to achieve satisfactory detection performance, a large amount of training samples with the corresponding labels is essential. In [10], it was found that 10^6 training samples are sufficient to achieve the near-optimal BER performance. However, collecting a large amount of training data is costly, time-consuming, and consumes additional power. A large training data size will also increase the training complexity. Furthermore, when the channel raw BER is above a certain value, the decoding of ECC will fail. As a result, it may not be feasible to obtain enough correct labels through ECC decoding.

Fig. 3 illustrates the influence of the number of training samples on the RBER performance. The optimal BER which assumes that perfect channel knowledge is known to the detector (derived in Section IV-F) is also included as the benchmark. Observe that the RNN detector can achieve the optimum performance when the number of training samples $N_{\text{train}} = 1 \times 10^6$. When N_{train} decreases, the BER performance degrades. In the next section, novel DTL-based detection approaches will be presented to reduce the number of training samples and labels without performance degradation.

IV. TRANSFER LEARNING-BASED DATA DETECTION

In this section, we first formulate the data detection for the flash memory channel as a TL problem, and then propose a model-based TL algorithm to effectively reduce the number of required training samples and labels. We further propose a UDA-based TL algorithm to detect the data without any labels in the target domain. It can work well for the scenarios where the ECC decoding fails due to unknown channel offset/noise.

A. Formulation of TL Problem

We first define a domain $\mathcal{D}$, which consists of features $\boldsymbol{v}$ and labels $\boldsymbol{x}$, i.e., $\mathcal{D} = \{v_i, x_i\}_{i=1}^n$, where n is the number of samples in the domain. To enable TL, we define two domains, namely, the source domain $\mathcal{D}_s$ and the target domain $\mathcal{D}_t$. Generally, the aim of TL is to transfer the knowledge from the source domain to the target domain, thus improving the performance of the intended task.

In our case, the source domain $\mathcal{D}_s$ consists of $\boldsymbol{v}$ and $\boldsymbol{x}$ at $N_{\text{PE}} = 0$ and $T = 0$. It is obvious that we have sufficient training data and labels at the source domain since they are easy to be obtained when the flash memory channel is not severely corrupted by noise in the beginning of its life. The target domain $\mathcal{D}_t$ consists of data and labels at $N_{\text{PE}} = N_{\text{PE}}^{\text{target}}$ and $T = T^{\text{target}}$, where $N_{\text{PE}}^{\text{target}}$ and T^{target} are the targeted number of P/E cycles and retention time while performing data detection. Usually, the training samples and labels are limited in the target domain, due to the restriction of read latency, power consumption, and ECC capabilities. However, it is noticed that the source domain and target domain have similar channel characteristics. For example, they have the same number of threshold voltage states, and the statistical distributions of threshold voltages of these states are of the same type (in this work, we follow the literature work and assume the distributions are Gaussian), although the respective values of mean and variance differ. This allows us to apply the DTL technique for the data detection at the target domain with significantly reduced number of training samples and labels.

Some properties of the source domain and target domain for our TL problem are given as follows.

1) The features of the source domain $\boldsymbol{v}_s = \{v_{s,1}, v_{s,2}, \ldots, v_{s,n_s}\}$, where $v_{s,k} \in \mathbb{R}$ and n_s is the number of samples in the source domain. The features of the target domain $\boldsymbol{v}_t = \{v_{t,1}, v_{t,2}, \ldots, v_{t,n_t}\}$, where $v_{t,k} \in \mathbb{R}$, and n_t is the number of samples in the target domain. The feature space of the source domain $\mathcal{V}_s$ is the same as that of the target domain $\mathcal{V}_t$, i.e., $\mathcal{V}_s = \mathcal{V}_t$.
2) The labels of the source domain $\boldsymbol{x}_s = \{x_{s,1}, x_{s,2}, \ldots, x_{s,n_s}\}$, where $x_{s,k} \in \{0, 1, \ldots, 2^q - 1\}$. The labels of the target domain $\boldsymbol{x}_t = \{x_{t,1}, x_{t,2}, \ldots, x_{t,n_t}\}$, where $x_{t,k} \in \{0, 1, \ldots, 2^q - 1\}$. The label space of the source domain $\mathcal{X}_s$ is the same as that of the target domain $\mathcal{X}_t$, i.e., $\mathcal{X}_s = \mathcal{X}_t$.
3) The conditional PDF $p(\boldsymbol{v}_s|\boldsymbol{x}_s)$ of the source domain is different from $p(\boldsymbol{v}_t|\boldsymbol{x}_t)$ of the target domain, i.e., $p(\boldsymbol{v}_s|\boldsymbol{x}_s) \neq p(\boldsymbol{v}_t|\boldsymbol{x}_t)$.

According to above properties, the TL problem in our case can be classified as the homogeneous TL. The task of the TL is to transfer the knowledge from the source domain to the target domain, and learn a prediction function $f : \boldsymbol{v}_t \rightarrow \boldsymbol{x}_t$ in the target domain to minimize the loss ϕ between the predicted outputs $f(\boldsymbol{v}_t)$ and labels $\boldsymbol{x}_t$:

$$f^* = \arg\min_f \mathbb{E}_{(\boldsymbol{v}_t, \boldsymbol{x}_t) \in \mathcal{D}_t} \phi(f(\boldsymbol{v}_t), \boldsymbol{x}_t), \tag{12}$$

where $\phi(f(\boldsymbol{v}_t), \boldsymbol{x}_t)$ is the loss function between $f(\boldsymbol{v}_t)$ and $\boldsymbol{x}_t$, which can be defined as the MSE or other loss functions.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

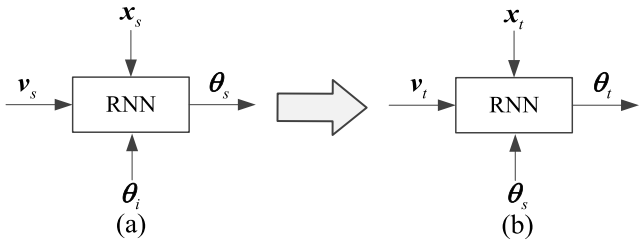


Fig. 4. Training process of model-based DTL (a) Pre-training (b) Finetuning (Retraining).

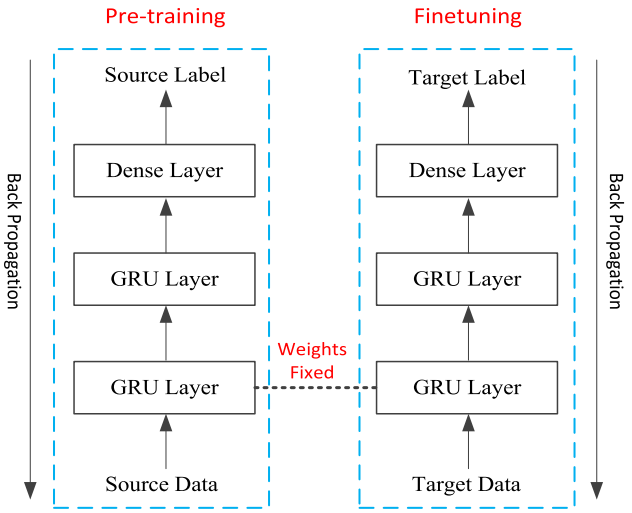


Fig. 5. Illustration of model-based DTL with weights reuse.

In the following, we present DTL methods to accomplish this task based on the RNN detection framework.

B. Model-Based DTL

We first combine DTL directly with the RNN and the resulting model-based DTL is realized by pre-training an RNN in the source domain first, and then transferring the trained RNN model and finetuning the network parameters in the target domain. Specifically, the RNN is trained in the source domain, and the updated set of network parameters $\boldsymbol{\theta}_s$ can obtained. Then, we train the same RNN in the target domain with initial parameters $\boldsymbol{\theta}_s$. Moreover, to reduce the training complexity, we further propose to freeze some parameters during training.

The proposed model-based DTL is illustrated by Fig. 4 and Fig. 5. First, the pre-trained model parameters $\boldsymbol{\theta}_s$ are obtained by training the RNN in the source domain with source data $\boldsymbol{v}_s$ and labels $\boldsymbol{x}_s$. In the target domain, the same RNN architecture is deployed and the model parameters are initialized by $\boldsymbol{\theta}_s$. Then, the RNN model is retrained by finetuning the parameters from the pre-trained model. Moreover, it has been demonstrated that in a deep neural network, the first few layers only learn general features and we can directly transfer them to new tasks [38]. Motivated by this, as shown in Fig. 5, we fix model weights of the first GRU layer. Only the second GRU layer and the fully-connected layer are retrained in the target domain, resulting in less training complexity. For example, if the input size of the RNN is 20, the number of training parameters

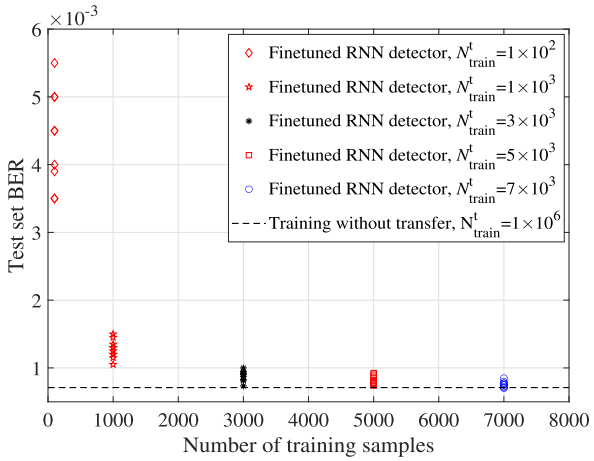


Fig. 6. The RBER performance of the finetuned RNN detector with different number of training samples in the target domain, with $N_{\text{PE}}^{\text{target}} = 5 \times 10^3$ and $T^{\text{target}} = 5 \times 10^3$ hours.

can be reduced from 3921 to 2541, which yields about 35% reduction of the training complexity. The steps of the proposed model-based DTL are summarized in **Algorithm 1**.

Algorithm 1 Model-Based DTL Detection

Input: Source data: $\boldsymbol{v}_s$, source labels: $\boldsymbol{x}_s$, target data for training: $\boldsymbol{v}_t^{\text{train}}$, target labels for training: $\boldsymbol{x}_t^{\text{train}}$, target data for testing: $\boldsymbol{v}_t^{\text{test}}$.

Output: detected symbols in the target domain: $\boldsymbol{x}_t$

`Training Stage`

1: With $\boldsymbol{v}_s$ and $\boldsymbol{x}_s$, train the RNN to obtain model parameters $\boldsymbol{\theta}_s$.

2: Initialize the RNN model with $\boldsymbol{\theta}_s$, and freeze the weights of the first GRU layer of the RNN model.

3: Finetune the RNN model to obtain parameters $\boldsymbol{\theta}_t$ by training the RNN with with $\boldsymbol{v}_t$ and $\boldsymbol{x}_t$.

`Testing Stage`

4: Detect the target data $\boldsymbol{v}_t^{\text{test}}$ using the RNN with $\boldsymbol{\theta}_t$ to obtain $\boldsymbol{x}_t$.

To investigate the influence of finetuning on the performance in the target domain, the RBERs of our proposed model-based DTL for MLC flash memory are illustrated by Fig. 6. In the source domain, 1×10^6 training samples and labels are used to pre-train the model. In the target domain, we vary the number of training samples $N_{\text{train}}^{\text{t}}$. For each case, to evaluate the stability of our approach, we take 10 trials of RNN finetuning, where the training data are generated randomly for each trial. Observe that as the training data size increases, the BER of the RNN detector becomes more stable, and converges to the optimum. Moreover, with only 7×10^3 training samples in the target domain, performance of the proposed model-based DTL can closely approach the best performance where the training with 1×10^6 samples is conducted directly in the target domain.

This result indicates that the proposed model-based DTL approach can significantly reduce the training data size by two orders of magnitude. This is due to the main channel characteristics (as described in Section II) of the target domain and the source domain are similar, and thereby it is easier for

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

RNN to learn based on the knowledge also learned from the source domain. Therefore, starting from the pre-trained model parameters, the training process in the target domain can be significantly accelerated.

C. UDA-Based DTL

Although the above described model-based DTL is simple and can achieve excellent performance by finetuning the pre-trained model parameters, it still requires a certain number of labels in the target domain. However, when the channel offset is large (*e.g.* caused by a long retention time or a large number of P/E cycles), the decoding of ECC may fail. Then it will be difficult to obtain enough labels for training the neural network.

As a type of feature-based TL, UDA methods migrate knowledge from a labeled source domain to an unlabeled target domain [29]. Popular UDA approaches include domain alignment with statistic divergence, adversarial learning and so on [28]. In this section, we propose a DTL algorithm based on UDA, such that no labels are required in the target domain. Specifically, we will align the mean of each voltage state between the source domain and the target domain. However, it is difficult to directly calculate the mean of each threshold voltage state in the target domain without any labels due to the overlapping of voltage distributions of different states and the unknown channel offset. To solve this problem, we adopt a K-means clustering approach to find the mean of each threshold voltage state.

The K-means clustering algorithm aims to partition the n read-back voltages into K clusters $C = \{C_0, C_1, \ldots, C_{K-1}\}$ [39], such that the intra-cluster distances are minimized. Hence, the optimized clusters C^* are given by

$$C^* = \arg\min_{C} \sum_{i=0}^{K-1} \sum_{v_j \in C_i} (v_j - \mu_{t,i})^2, \tag{13}$$

where $\mu_{t,i}$ is the mean of the i-th cluster in the target domain, and $K = 2^q$ since there are 2^q voltage states. The K-means clustering is an iterative algorithm to find the solution of (13). At the beginning of the algorithm, the initial centroid of each cluster needs to be determined and it will affect the accuracy and convergence speed of the K-means clustering algorithm. Let the order of centroids at the k-th iteration follows

$$\mu_{t,0}^{(k)} < \mu_{t,1}^{(k)} < \cdots < \mu_{t,2^q-1}^{(k)}. \tag{14}$$

In our case, it is natural to initialize the centroid of each cluster as

$$\mu_{t,i}^{(0)} = V_{s_i}, i = 0, 1, \ldots, 2^q - 1, \tag{15}$$

since they are the means of initial voltage states. Then, the algorithm proceeds by iterating the following two steps:

1) **Assignment step** In the t-th iteration, each v_j, $j = 1, 2, \ldots, n$ is assigned to the nearest cluster according to the following equation:

$$i^* = \arg\min_{i} (v_j - \mu_{t,i}^{(k)})^2. \tag{16}$$

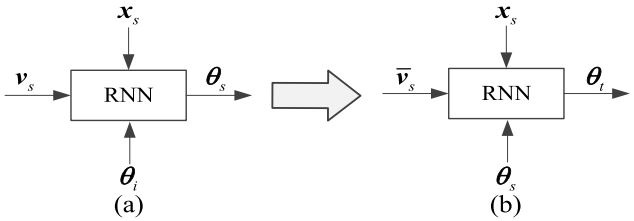


Fig. 7. Training process of UDA-based DTL (a) Pre-training (b) Finetuning (Retraining).

Then, v_j is assigned to cluster C_{i^*}, $i^* \in \{0, 1, \ldots, 2^q - 1\}$.

2) **Updating step** After all v_j's are assigned to the corresponding clusters, the mean of each cluster is updated as

$$\mu_{t,i}^{(k+1)} = \frac{1}{|C_i|} \sum_{v_j \in C_i} v_j, \tag{17}$$

where $|C_i|$ is the number of elements in C_i.

The above iterations are stopped when the mean of each cluster does not change or the maximum number of iterations is reached. By applying this clustering algorithm, we can estimate the mean $\mu_{t,i}$, $i = 0, 1, \ldots, 2^q - 1$, of the i-th voltage state in the target domain without any labels.

With $\mu_{t,i}$, we can align the mean of the sub-domain source data as

$$\bar{\boldsymbol{v}}_{s,i} = \boldsymbol{v}_{s,i} - \mu_{s,i} + \mu_{t,i}, \tag{18}$$

where $\boldsymbol{v}_{s,i}$ is the source data that is associated with label i, and $\mu_{s,i}$ is the mean of these data. Note that the training data and labels in the source domain are known, and hence $\boldsymbol{v}_{s,i}$ and $\mu_{s,i}$ can be easily obtained. With (18), the mean of each voltage state in the source domain can be aligned with the target domain. Note that we did not align the variance of the source data with that of the target data as it is difficult to accurately obtain the variance of each voltage state without any labels.

After the above UDA process, as shown in Fig. 7, we can finetune the RNN with the transformed source data $\bar{\boldsymbol{v}}_s$ and the original source labels $\boldsymbol{x}_s$ to obtain updated model parameters $\boldsymbol{\theta}_t$. Moreover, to accelerate the convergence speed and reduce the training complexity, we combine the UDA with the model-based DTL described in Section IV-B. Meanwhile, we can also retrain the pre-trained RNN by fixing some layers and finetuning the remaining parameters. We can then use this updated RNN to detect data in the target domain directly. The detailed procedure of our proposed UDA-based DTL is described in **Algorithm 2**.

Compared with model-based DTL, the UDA-based DTL does not require any labels in the target domain with the cost of additional DA process. During the UDA process, the most time-consuming step is the K-means clustering with a complexity of $O(2^q n_t I)$, where n_t is the number of data samples in $\boldsymbol{v}_t$, and I is the number of iterations. According to our simulations, with initial centroids given by (15), the proposed K-means clustering algorithm for MLC, TLC and QLC can converge with 3, 40, and 150 iterations on average,

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

Algorithm 2 UDA-Based DTL Detection

Input: Source data: $\boldsymbol{v}_s$, source labels: $\boldsymbol{x}_s$, target data: $\boldsymbol{v}_t$, the initial RNN model.

Output: Trained RNN parameters: $\boldsymbol{\theta}_t$, detected symbols in the target domain: $\boldsymbol{x}_t$

`Training Stage`

1: With $\boldsymbol{v}_s$ and $\boldsymbol{x}_s$, follow model-based DTL to pre-train the RNN to obtain network parameters $\boldsymbol{\theta}_s$.

2: Calculate $\bar{\boldsymbol{v}}_{s,i}$, $i = 0, 1, \ldots, 2^q - 1$ with (18) to align the sub-domain mean of the source data with that of the target data.

3: Initialize the RNN with $\boldsymbol{\theta}_s$, and retrain the RNN with $\bar{\boldsymbol{v}}_s$ and $\boldsymbol{x}_s$ to obtain $\boldsymbol{\theta}_t$.

`Testing Stage`

4: Detect $\boldsymbol{v}_t$ by the trained-RNN with $\boldsymbol{\theta}_t$ to obtain $\boldsymbol{x}_t$.

respectively. Note that although the number of iterations will increase with q increases, the maximum number of q is 4, which is QLC flash memory. Hence, the number of iterations will not be very large.

D. RNNA Threshold Detection With DTL

Although the proposed DTL-based RNN detectors is effective, it is not practical to use the RNN to detect every data block, as it will incur significant read latency and power consumption. To avoid activating the RNN for each data block, we can derive read thresholds from the RNN detected data and then directly use the updated read thresholds for subsequent data detection. We name the corresponding detection RNNA threshold detection with DTL.

Both the model-based and UDA-based DTL can be used to derive the updated read thresholds for data detection. In particular, for given $\boldsymbol{v}$ and a set of hard-decision read thresholds $\left\{V_1^{\text{th}}, V_2^{\text{th}}, \ldots, V_{2^q-1}^{\text{th}}\right\}$, we can obtain the detected symbols $\bar{\boldsymbol{x}}$. Meanwhile, given $\boldsymbol{v}$, the RNN can also output its estimated symbols $\tilde{\boldsymbol{x}}$. Hence, the RNN learned read thresholds can be obtained by searching for the read thresholds $\left\{V_1^{\text{opt}}, V_2^{\text{opt}}, \ldots, V_{2^q-1}^{\text{th}}\right\}$ that minimize the Hamming distance between $\bar{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}$:

$$\left\{V_1^{\text{opt}}, V_2^{\text{opt}}, \ldots, V_{2^q-1}^{\text{opt}}\right\} = \underset{\left\{V_1^{\text{th}}, V_2^{\text{th}}, \ldots, V_{2^q-1}^{\text{th}}\right\}}{\arg\min} d(\bar{\boldsymbol{x}}, \tilde{\boldsymbol{x}}). \quad (19)$$

To obtain $\left\{V_1^{\text{opt}}, V_2^{\text{opt}}, \ldots, V_{2^q-1}^{\text{opt}}\right\}$ efficiently, we first uniformly quantize the search space into m intervals (typical value of $m \geq 100$), with boundaries $b_0, b_1, \ldots, b_m$, where $b_0 = -\infty < b_1 < \cdots < b_{m-1} < b_m = \infty$. Then, the problem becomes finding $2^q - 1$ thresholds from $b_0, b_1, \ldots, b_m$, such that the Hamming distance between $\bar{\boldsymbol{x}}$ and $\tilde{\boldsymbol{x}}$ is minimized. For MLC flash memory, $2^q - 1 = 3$ and for TLC flash memory, $2^q - 1 = 7$. To solve this problem, we can adopt a dynamic programming (DP) approach with complexity $\mathcal{O}((2^q - 1)m^2)$, and the details can be found in [10]. After obtaining the updated read thresholds, we can use them directly for data detection. As will be shown in Section V, the read thresholds learned from DTL can achieve near-optimal detection BER performance.

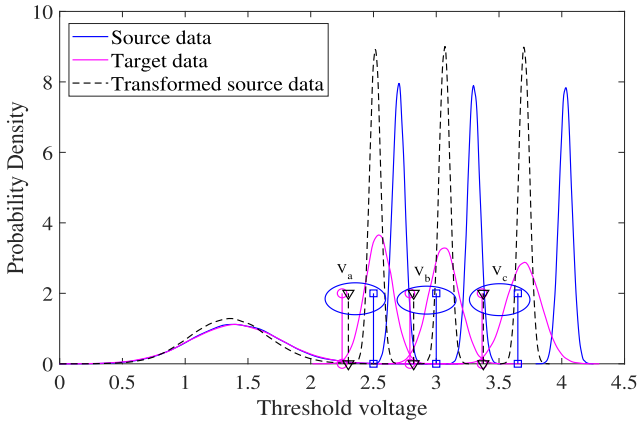


Fig. 8. The PDFs and learned read thresholds of the source data, target data, and transformed source data for MLC flash memory with $N_{\text{PE}}^{\text{target}} = 10^4$ and $T^{\text{target}} = 10^4$ hours (read thresholds markers: square for source data, circle for target data, triangle for transformed source data).

Fig. 8 shows the PDFs of the source data, target data, and transformed source data, with the associated read thresholds, with $N_{\text{PE}}^{\text{test}} = 10^4$ and $T^{\text{test}} = 10^4$ for the MLC flash memory. The read thresholds of the target data and transformed source data are derived according to (19). Observe that although the PDF of the transformed source data is more similar to that of the source data than the target data, the read thresholds learned from the transformed source data and the target data are quite close. This indicates that the RNN trained by the transformed source data can achieve a good detection performance. On the other hand, the read thresholds based on the source data are far from that based on the target data, and hence will lead to a severe performance degradation.

E. UDA-Based Threshold Detection With Original Read Thresholds

Inspired by the threshold detector with the UDA-based DTL described above, we further propose a simple UDA-based threshold detection scheme, for which the neural network is not needed in the target domain and we can directly use the original read thresholds in the source domain to perform detection.

That is, by employing the K-means clustering algorithm described in Section IV-C in the target domain, we can estimate the mean $\mu_{t,i}$ of the i-th voltage state. However, instead of updating the source data in the UDA-based DTL, we update the target data, given by

$$\bar{\boldsymbol{v}}_{t,i} = \tilde{\boldsymbol{v}}_{t,i} - \mu_{t,i} + \mu_{s,i}, \quad (20)$$

where $\tilde{\boldsymbol{v}}_{t,i}$ can be obtained by using the pseudo labels obtained from the K-means clustering. With (20), the mean of each voltage state in the target domain is aligned with that in the source domain. Then, these target data can be detected by using the optimal read thresholds $\left\{V_1^s, V_2^s, \ldots, V_{2^q-1}^s\right\}$ in the source domain.

In this way, the RNN in the target domain is no longer needed for the data detection. Moreover, in practical flash memories, the initial read thresholds $\left\{V_1^s, V_2^s, \ldots, V_{2^q-1}^s\right\}$ have already been provided by manufacturers. Therefore, the RNN in the source domain is not needed as well. The details

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

Algorithm 3 UDA-Based Threshold Detection

Input: Source data: $\boldsymbol{v}_s$, source labels: $\boldsymbol{x}_s$, source domain read thresholds: $V_1^s, V_2^s, \ldots, V_{2^q-1}^s$, target data: $\boldsymbol{v}_t$.

Output: Detected symbols in the target domain: $\boldsymbol{x}_t$

1: With $\boldsymbol{v}_s$ and $\boldsymbol{x}_s$, calculate the mean of each voltage state for the source data, given by $\mu_{s,i}, i = 0, 1, \ldots, 2^q - 1$.
2: With $\boldsymbol{v}_t$, find the mean of each voltage state for the target data using K-means clustering algorithm in Section IV-C.
3: Calculate $\bar{\boldsymbol{v}}_{t,i}$, $i = 0, 1, \ldots, 2^q - 1$ with (20) to align the sub-domain mean of the target data with that of the source data.
4: Detect the transformed target data $\bar{\boldsymbol{v}}_t$ with read thresholds $\{V_1^s, V_2^s, \ldots, V_{2^q-1}^s\}$ to obtain $\boldsymbol{x}_t$.

of our proposed UDA-based threshold detection is given in **Algorithm 3**.

F. Uncoded Error Rate Analysis

In this subsection, we derive the the optimum uncoded SER and BER by assuming the channel knowledge is perfectly known. They serve as benchmarks of the various detectors proposed earlier. For given hard-decision read thresholds $\{V_1^{\text{th}}, V_2^{\text{th}}, \ldots, V_{2^q-1}^{\text{th}}\}$, assuming that the memory cells are programmed into 2^q voltage states with equal probabilities, the SER is calculated as

$$\begin{aligned} P_s &= \sum_{i=0}^{2^q-1} P(v_{s_i}) P(e|v_{s_i}) \\ &= \frac{1}{2^q}\Bigg(P(v > V_1^{\text{th}}|v_{s_0}) + \sum_{i=1}^{2^q-2} P(v < V_i^{\text{th}} \cup v > V_{i+1}^{\text{th}}|v_{s_i}) \\ &\quad + P(v < V_{2^q-1}^{\text{th}}|v_{s_{2^q-1}})\Bigg), \end{aligned} \tag{21}$$

where e denotes the event that an error occurs. With given PDF $p_{v_{s_i}}, i = 0, 1, \ldots, 2^q - 1$ for each voltage state v_{s_i}, the final expression of the SER is given by

$$\begin{aligned} P_s = \frac{1}{2^q}\Bigg(&\int_{V_1^{\text{th}}}^{\infty} p_{v_{s_0}} dv + \sum_{i=1}^{2^q-2}\Bigg(\int_{-\infty}^{V_i^{\text{th}}} p_{v_{s_i}} dv \\ &+ \int_{V_{i+1}^{\text{th}}}^{\infty} p_{v_{s_i}} dv\Bigg) + \int_{-\infty}^{V_{2^q-1}^{\text{th}}} p_{v_{s_{2^q-1}}} dv\Bigg). \end{aligned} \tag{22}$$

By searching read thresholds $\{V_1^{\text{th}}, V_2^{\text{th}}, \ldots, V_{2^q-1}^{\text{th}}\}$ that minimize (22), we can obtain the optimum SER, which can serve as the lower bound of the proposed detectors. Next, the corresponding BER can be estimated from the SER. Note that when the Gray mapping is adopted, the adjacent voltage states only differ by one bit. Hence, assuming that only the adjacent states dominate the errors, the BER can be estimated as

$$P_b \approx P_s/q. \tag{23}$$

Note that (23) is a lower bound of the exact BER, since it only considers errors occurred in the adjacent states. However, for the case of TLC where intervals between adjacent voltage levels are getting smaller than the MLC case, more states will affect the BER. Hence, (23) is not accurate enough. To obtain a more accurate BER estimation, we consider adjacent states that differ by either 1-bit or 2-bit. Therefore, the BER is given by

$$P_b \approx \sum_{i=0}^{2^q-1} P(v_{s_i})\left(\frac{1}{q}P(e_1|v_{s_i}) + \frac{2}{q}P(e_2|v_{s_i})\right), \tag{24}$$

where e_1 denotes the event that an error occurs in the adjacent states, while e_2 represents the event that an error occurs in the non-adjacent states which results in 2-bit errors per symbol. In (24), when $i = 0$, $P(e_1|v_{s_i})$ and $P(e_2|v_{s_i})$ are given by

$$\begin{aligned} P(e_1|v_{s_i}) &= P(V_1^{\text{th}} < v < V_2^{\text{th}}|v_{s_0}) \\ &= \int_{V_1^{\text{th}}}^{V_2^{\text{th}}} p_{v_{s_0}} dv \end{aligned} \tag{25}$$

and

$$\begin{aligned} P(e_2|v_{s_i}) &= P(v > V_2^{\text{th}}|v_{s_0}) \\ &= \int_{V_2^{\text{th}}}^{\infty} p_{v_{s_0}} dv. \end{aligned} \tag{26}$$

Similarly, for $i = 1, 2, \ldots, 2^q - 1$, $P(e_1|v_{s_i})$ and $P(e_2|v_{s_i})$ can be calculated. By substituting $P(e_1|v_{s_i})$ and $P(e_2|v_{s_i})$ into (24), a more accurate BER estimation can be obtained.

As illustrated by Fig. 9, (22) and (24) can match well with the simulated SERs and BERs for the TLC case, while (23) slightly underestimate the BER when the number of P/E cycles is less than 1000. Therefore, (22) and (24) can serve as references for the SER and BER of the proposed detectors, respectively.

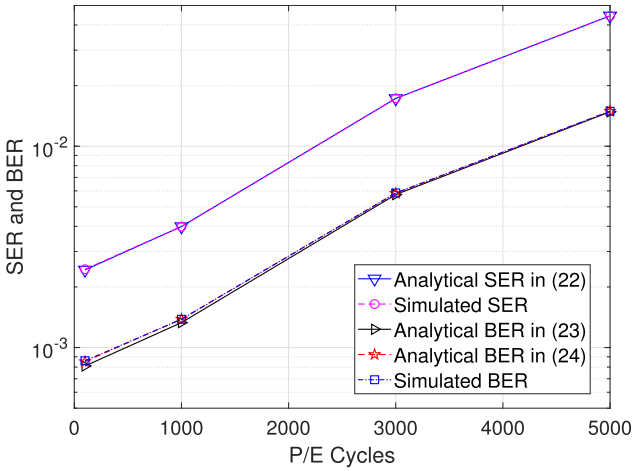


Fig. 9. The analytical and simulated SER and RBER performance of TLC flash memory at retention time 1×10^4 hours.

G. Discussions

1) Comparison of Proposed Detectors: Comparing the proposed three detection schemes, the RNNA threshold detection with model-based DTL (presented in Section IV-D) is supervised, while the RNNA threshold detection with UDA-based DTL (presented in Section IV-D) as well as the UDA-based threshold detection (presented in Section IV-E) are unsupervised. Therefore, these two detection schemes based on

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

TABLE II
THE COMPLEXITY OF DTL-BASED DETECTORS FOR FLASH MEMORIES

	Model-based DTL detector	UDA-based DTL detector	UDA-based threshold detector
RNN Training	$\mathcal{O}(\frac{n_t}{N}SL(D+L))$	$\mathcal{O}(\frac{n_t}{N}SL(D+L))$	0
K-means Clustering	0	$\mathcal{O}(2^q n_t I)$	$\mathcal{O}(2^q n_t I)$
Read Thresholds Derivation	$\mathcal{O}((2^q-1)m^2)$	$\mathcal{O}((2^q-1)m^2)$	0

UDA are more suitable for severe channel conditions where labels are not available. On the other hand, compared with the RNNA threshold detection with model-based DTL, these detection schemes require additional DA steps to approach the performance with the model-based DTL.

Unlike the RNNA threshold detection with either the model-based DTL or the UDA-based DTL, the UDA-based threshold detection does not require the neural network in both the source domain and the target domain. Therefore, it is simple to implement. However, it needs to update each target domain data using (20) before detection, which will incur additional latency. On the other hand, the RNNA threshold detection with the model-based DTL or the UDA-based DTL can directly use the updated read thresholds to detect data. Therefore, the proposed three knowledge transfer-based detection schemes should be applied according to the availability of labels and system requirements.

2) Complexity Analysis: In this part, the complexity of our proposed DTL-based detectors is analyzed. First, the proposed RNN architecture consists of two GRU layers and one output layer. For the GRU layer with L GRU cells and input dimension D, the number of parameters is $3L(D+L+1)$. In our proposed RNN architecture, N readback threshold voltages are fed into the network for each training or inference. Hence, given a total of n_t training samples in the target domain and S training epochs, the training complexity can be approximated by $O(\frac{n_t}{N}SL(D+L))$. The inference complexity can be estimated by reducing the number of epochs to one. In this work, $L=20$ and $N=20$. We have $D=1, 20$ for the first and second GRU layers, respectively. Second, the complexity of K-means clustering for the UDA-based DTL detection and UDA-based threshold detection is given in Section IV-C. Third, the complexity for deriving read thresholds is given in Section IV-D, which is $\mathcal{O}((2^q-1)m^2)$. To obtain good performance, the typical value of m is 100 or larger. Hence, the overall complexity for different detection schemes in the target domain is summarized in Table II.

First, it can be seen from Table II that the training complexity is linearly proportional to the number of training samples. Since it has been demonstrated in Section IV-B that compared with the DL-based approach [10], the proposed DTL algorithms can reduce the number of training samples and labels in the target domain by two orders of magnitude, the training complexity is also decreased by two orders of magnitude. Second, as described in Section IV-C, the maximum value of q is 4, thereby we conclude that the overall complexity will be dominated by the RNN training.

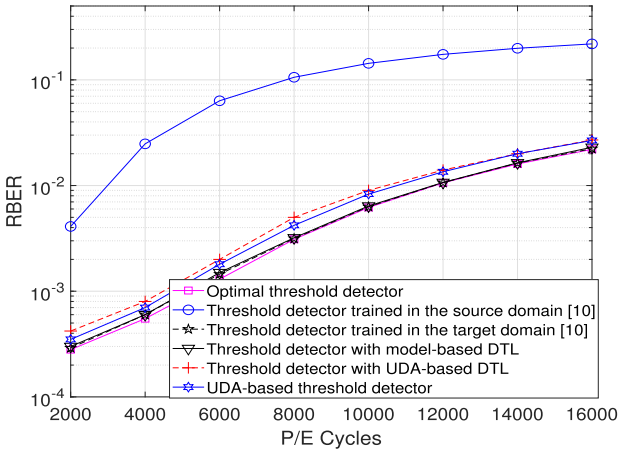


Fig. 10. The RBER performance of different threshold detectors for MLC flash memory at $T^{\text{target}} = 1.2 \times 10^4$ hours.

V. NUMERICAL AND SIMULATION RESULTS

In this work, the RNN-based DTL algorithms are implemented with Pytorch DL framework [40]. In the following, the channel RBER and the LDPC-coded BER for MLC and TLC flash memory channels are evaluated with our proposed threshold detectors. In our experiments, the number of training samples and labels in the target domain is 10^6 for the RNNA threshold detector trained without TL (presented in Section III-B), while it is taken to be 10^4 for the RNNA threshold detectors with DTL. In particular, for the RNNA threshold detector with UDA-based DTL, labels in the target domain are not needed. We evaluate both the uncoded RBER and the LDPC-coded BER performance of various threshold detectors, for both the MLC and TLC flash memories. Note that for the MLC case, we take $V_{s_0\sim s_3} = \{1.4, 2.6, 3.2, 3.93\}$, while for the TLC case, we set $V_{s_0\sim s_7} = \{1.4, 2.2, 2.6, 3.0, 3.4, 3.8, 4.2, 4.6\}$.

A. Raw BER Performance

Figs. 10 and 11 illustrate the RBER performance of different RNNA threshold detectors for the MLC flash memory over different P/E cycles and retention time. The BER performance of the optimum threshold detector given by (24) is also included as a reference. First, we observe a large performance degradation of the RNNA threshold detector trained in the source domain, and it is due to the channel mismatch between the source domain and the target domain. This indicates the necessity of utilizing the knowledge of the target domain to improve the detection performance. Second, it can be observed that the RNNA threshold detector with

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

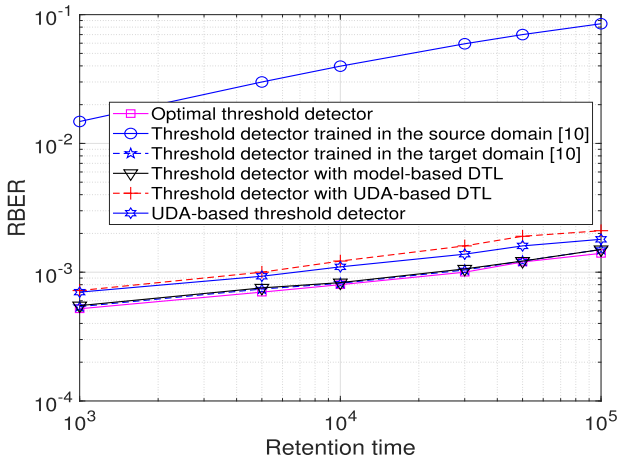


Fig. 11. The RBER performance of different threshold detectors for MLC flash memory at $N_{\text{PE}}^{\text{target}} = 5 \times 10^3$.

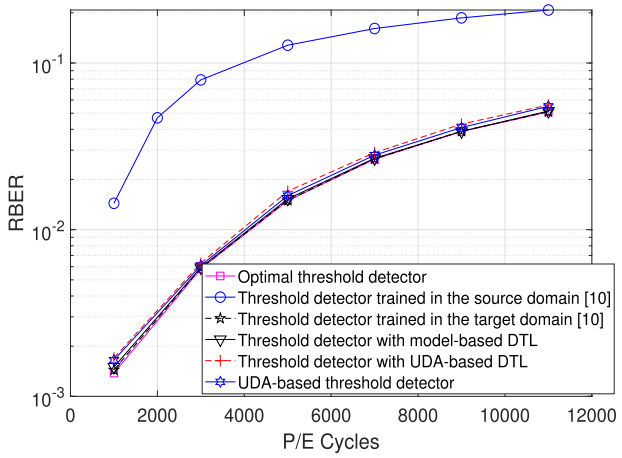


Fig. 12. The RBER performance of different threshold detectors for TLC flash memory at $T^{\text{target}} = 1 \times 10^4$ hours.

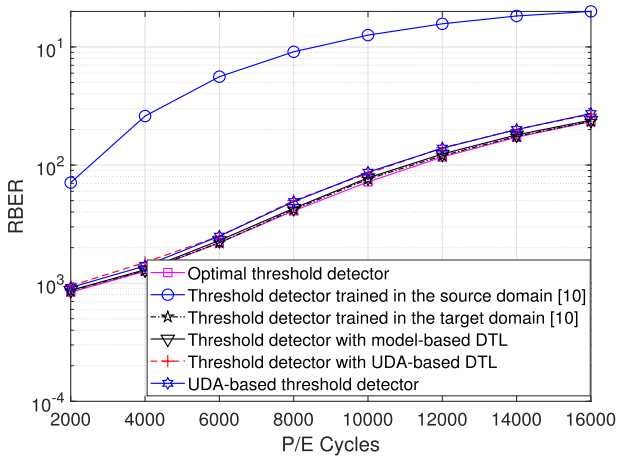


Fig. 13. The RBER performance of different threshold detectors for MLC flash memory with Gamma distributed noises at $T^{\text{target}} = 1.2 \times 10^4$ hours.

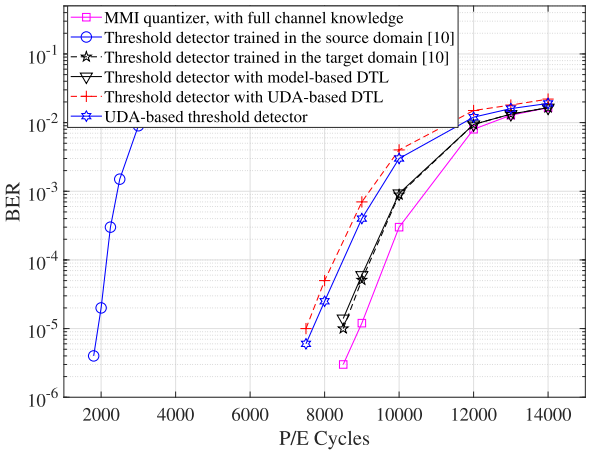


Fig. 14. The BER performance of the LDPC code with different threshold detectors for MLC flash memory at $T^{\text{target}} = 1.2 \times 10^4$ hours.

the proposed model-based DTL achieves the performance of the RNNA threshold detector directly trained in the target domain. The RBERs of the two detectors also approach those of the optimum threshold detector. Third, performance of both the RNNA threshold detector with UDA-based DTL and the UDA-based threshold detector is slightly worse than the optimum performance. The reason is that only the mean of each voltage state is aligned between the source domain and the target domain in these detectors. As the variance of each voltage state is not aligned, there will be a threshold voltage PDF mismatch between the two domains.

The RBERs of different threshold detectors for the TLC flash memory is shown in Fig. 12. Different from the results for the MLC case, it can be seen that for the TLC flash memory, the performance of both the RNNA threshold detector with UDA-based DTL and the UDA-based threshold detector closely approaches the optimum performance. The reason for this interesting result is that the TLC flash memory channel is more "symmetric" than the MLC flash memory channel. More specifically, the TLC flash memory cell has more threshold voltage states than the MLC one. Except for the erase state, the variances of all the programmed states are close to each other. This implies that the TLC flash memory has a higher percentage of voltage states with similar variances than the MLC flash memory. The proposed UDA is based on the mean of each voltage state, while the variance is not aligned. Hence, the proposed method is better for the TLC case, leading to better performance of the threshold detection schemes using UDA.

Finally, to further verify the effectiveness of our proposed DTL approaches for different noise distributions, we consider the case that the noises in the target domain follow a Gamma distribution, while the noise PDF in the source domain remains Gaussian. It can be seen from Fig. 13 that although the noise distributions of the source domain and target domain are different, our proposed DTL-based detectors can still achieve near-optimal performance. This indicates that the stacked RNN in Fig. 2 is capable of learning the detection of different noise distributions, and the proposed DTL approaches can effectively transfer the knowledge from the source domain to the target domain even if they have different noise PDFs. To conclude, Figs. 10-13 indicate that the proposed threshold detectors can effectively combat the unknown channel offset caused by the P/E cycling and data retention.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

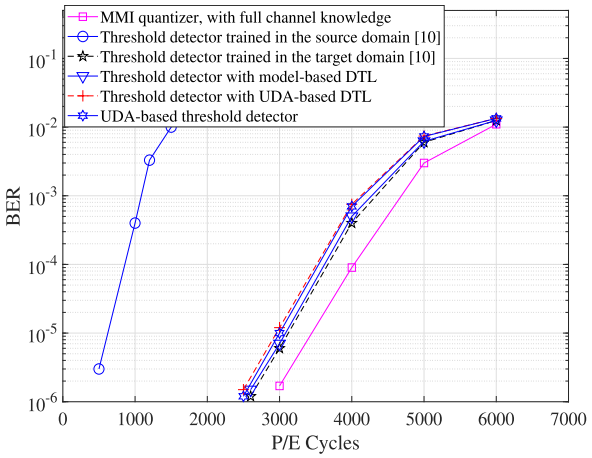


Fig. 15. The BER performance of the LDPC code with different threshold detectors for TLC flash memory at $T^{\text{target}} = 1 \times 10^3$ hours.

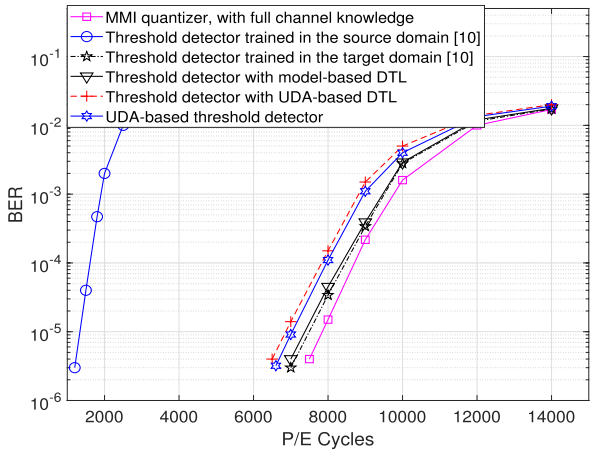


Fig. 16. The BER performance of the LDPC code with different threshold detectors for MLC flash memory with Gamma distributed noises at $T^{\text{target}} = 1.2 \times 10^4$ hours.

B. Coded BER Performance

To examine the decoding performance of ECCs with learned thresholds, an irregular LDPC code with codeword length of 4544 bits and information length 4096 bits [8] is employed. The degree distribution of this code is given by

$$\lambda(x) = 0.0682x + 0.1822x^2 + 0.1329x^3 + 0.6167x^4,$$
$$\rho(x) = 0.22x^{38} + 0.78x^{39}. \tag{27}$$

This LDPC code is constructed by the progressive edge-growth algorithm [41]. The decoding algorithm is the normalized min-sum (NMS) algorithm [42] with at most 20 decoding iterations. For the threshold detectors, the initial LLR is set to 5 or -5, depending on the detected bit being 0 or 1, respectively. Furthermore, as a benchmark of the coded BER performance, we design the read thresholds by MMI of the quantized flash memory channel [7], which is based on the ideal assumption that the perfect channel knowledge is known. We use the corresponding BERs as the performance benchmark.

Figs. 14 and 15 present the BER performance of the LDPC codes with the MMI quantizer and our proposed threshold detectors over different P/E cycles, for the MLC and the TLC flash memories, respectively. Similar to the RBER performance shown in Fig. 10, Fig. 14 shows that for the MLC case, the BER performance of the RNNA threshold detector trained in the source domain is much worse than that of other detectors. The LDPC-coded BER performance using the RNNA threshold detector with the model-based DTL can achieve that of the RNNA threshold detector that is directly trained in the target domain, with much less training data. The performance of the RNNA threshold detector with UDA-based DTL and the UDA-based threshold detector is slightly worse than that with model-based DTL, which is also consistent with our RBER performance shown in Fig. 10. Finally, it can be seen that the performance of our proposed threshold detectors is slightly worse than that of the MMI quantizer, due to the lack of channel PDF.

However, for the TLC flash memory, Fig. 15 shows that the performance of both the RNNA threshold detector with UDA-based DTL and the UDA-based threshold detector can almost achieve that of the RNNA threshold detector that is directly trained in the target domain. This is again consistent with the uncoded case illustrated by Fig. 12. Furthermore, the LDPC-coded BER performance under non-Gaussian Gamma distributed noises in the target domain is illustrated in Fig. 16. It shows that all the proposed detectors can achieve near-optimal performance, and the UDA-based detectors perform slightly worse than the threshold detector with model-based DTL. Similar to Fig. 13, the LDPC-coded BER performance indicates that our proposed DTL-based detectors have the ability to transfer the knowledge to different noise distributions.

The simulation results illustrated by Figs. 10-16 demonstrate that our proposed threshold detectors based on the DTL and UDA can almost achieve the performance of the RNNA threshold detector that is directly trained in the target domain, with much less training data. This is because the proposed detection schemes have transferred the knowledge from the source domain to the target domain through model parameters and domain adaptation, which can achieve better error rate performance with much less training complexity.

VI. Conclusion

In this paper, we have formulated the data detection for the flash memory channel as a TL problem, and proposed a model-based DTL algorithm to effectively reduce the number of training samples and labels. We have further proposed a UDA-based DTL algorithm to cope with the scenarios where the channel has a large unknown offset such that the labels cannot be obtained. Furthermore, we have derived the RNNA threshold detectors with the proposed DTL algorithms. A UDA-based threshold detection scheme has also been proposed which completely avoids the use of neural network. Both the channel raw error rate analysis and simulation results demonstrate that our proposed DTL-based detection schemes can achieve the near-optimal BER performance with much less training data and/or without using any labels. A possible future research topic is the design and optimization of the multi-bit channel quantizer for the proposed DTL-based detectors.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] Y. Cai, S. Ghose, E. F. Haratsch, Y. Luo, and O. Mutlu, "Error characterization, mitigation, and recovery in flash-memory-based solid-state drives," *Proc. IEEE*, vol. 105, no. 9, pp. 1666–1704, Sep. 2017.

[2] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Error patterns in MLC NAND flash memory: Measurement, characterization, and analysis," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2012, pp. 521–526.

[3] H. Choi, W. Liu, and W. Sung, "VLSI implementation of BCH error correction for multilevel cell NAND flash memory," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 5, pp. 843–847, May 2010.

[4] K. Zhao, W. Zhao, H. Sun, X. Zhang, N. Zheng, and T. Zhang, "LDPC-in-SSD: Making advanced error correction codes work effectively in solid state drives," in *Proc. FAST*, Feb. 2013, pp. 243–256.

[5] J. Wang, T. Courtade, H. Shankar, and R. D. Wesel, "Soft information for LDPC decoding in flash: Mutual-information optimized quantization," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2011, pp. 1–6.

[6] G. Dong, N. Xie, and T. Zhang, "On the use of soft-decision error-correction codes in NAND flash memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 58, no. 2, pp. 429–439, Feb. 2011.

[7] J. Wang et al., "Enhanced precision through multiple reads for LDPC decoding in flash memories," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 5, pp. 880–891, May 2014.

[8] C. A. Aslam, Y. L. Guan, and K. Cai, "Read and write voltage signal optimization for multi-level-cell (MLC) NAND flash memory," *IEEE Trans. Commun.*, vol. 64, no. 4, pp. 1613–1623, Apr. 2016.

[9] Z. Mei, K. Cai, L. Shi, and X. He, "On channel quantization for spin-torque transfer magnetic random access memory," *IEEE Trans. Commun.*, vol. 67, no. 11, pp. 7526–7539, Nov. 2019.

[10] Z. Mei, K. Cai, and X. He, "Deep learning-aided dynamic read thresholds design for Multi-Level-Cell flash memories," *IEEE Trans. Commun.*, vol. 68, no. 5, pp. 2850–2862, May 2020.

[11] D.-h. Lee and W. Sung, "Estimation of NAND flash memory threshold voltage distribution for optimum soft-decision error correction," *IEEE Trans. Signal Process.*, vol. 61, no. 2, pp. 440–449, Jan. 2013.

[12] Y. Cai, E. F. Haratsch, O. Mutlu, and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2013, pp. 1285–1290.

[13] Y. Luo, S. Ghose, Y. Cai, E. F. Haratsch, and O. Mutlu, "Enabling accurate and practical online flash channel modeling for modern MLC NAND flash memory," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2294–2311, Sep. 2016.

[14] Y. Cai, Y. Luo, E. F. Haratsch, K. Mai, and O. Mutlu, "Data retention in MLC NAND flash memory: Characterization, optimization, and recovery," in *Proc. IEEE 21st Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2015, pp. 551–563.

[15] Z. Shi, Y. Fang, Y. Bu, and G. Han, "Convolutional neural network (CNN)-based detection for Multi-Level-Cell NAND flash memory," *IEEE Commun. Lett.*, vol. 25, no. 12, pp. 3883–3887, Dec. 2021.

[16] W. Wiriya and B. M. Kurkoski, "EM algorithm for DMC channel estimation in NAND flash memory," in *Proc. Non-Volatile Memories Workshop*, 2020, pp. 1–2.

[17] M. Sandell and A. Ismail, "Machine learning for LLR estimation in flash memory with LDPC codes," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 68, no. 2, pp. 792–796, Feb. 2021.

[18] F. Zhuang et al., "A comprehensive survey on transfer learning," *Proc. IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021.

[19] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.

[20] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang, "Domain adaptation via transfer component analysis," *IEEE Trans. Neural Netw.*, vol. 22, no. 2, pp. 199–210, Feb. 2011.

[21] B. Sun, J. Feng, and K. Saenko, "Return of frustratingly easy domain adaptation," *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1, Mar. 2016, pp. 2058–2065.

[22] J. Huang, A. Gretton, K. Borgwardt, B. Schölkopf, and A. Smola, "Correcting sample selection bias by unlabeled data," in *Proc. NIPS*, vol. 19, 2006, pp. 1–8.

[23] H. Daumé III, "Frustratingly easy domain adaptation," 2009, *arXiv:0907.1815*.

[24] M. Long, J. Wang, G. Ding, J. Sun, and P. S. Yu, "Transfer feature learning with joint distribution adaptation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2013, pp. 2200–2207,

[25] B. Sun and K. Saenko, "Deep CORAL: Correlation alignment for deep domain adaptation," in *Proc. Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands. Cham, Switzerland: Springer, Oct. 2016, pp. 443–450.

[26] Y. Zhu et al., "Deep subdomain adaptation network for image classification," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 4, pp. 1713–1722, Apr. 2021.

[27] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola, "Integrating structured biological data by kernel maximum mean discrepancy," *Bioinformatics*, vol. 22, no. 14, pp. e49–e57, Jul. 2006.

[28] X. Liu et al., "Deep unsupervised domain adaptation: A review of recent advances and perspectives," *APSIPA Trans. Signal Inf. Process.*, vol. 11, no. 1, pp. 1–51, 2022.

[29] G. Dong, N. Xie, and T. Zhang, "Enabling NAND flash memory use soft-decision error correction codes at minimal read latency overhead," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 60, no. 9, pp. 2412–2421, Sep. 2013.

[30] H. Wang, T.-Y. Chen, and R. D. Wesel, "Histogram-based flash channel estimation," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2015, pp. 283–288.

[31] G. Dong, Y. Pan, and T. Zhang, "Using lifetime-aware progressive programming to improve SLC NAND flash memory write endurance," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, no. 6, pp. 1270–1280, Jun. 2014.

[32] J.-D. Lee, S.-H. Hur, and J.-D. Choi, "Effects of floating-gate interference on NAND flash memory cell operation," *IEEE Electron Device Lett.*, vol. 23, no. 5, pp. 264–266, May 2002.

[33] T.-Y. Chen, A. R. Williamson, and R. D. Wesel, "Increasing flash memory lifetime by dynamic voltage allocation for constant mutual information," in *Proc. Inf. Theory Appl. Workshop (ITA)*, Feb. 2014, pp. 1–5.

[34] G. Dong, Y. Pan, N. Xie, C. Varanasi, and T. Zhang, "Estimating information-theoretical NAND flash memory storage capacity and its implication to memory system design space exploration," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 9, pp. 1705–1714, Sep. 2012.

[35] N. Mielke et al., "Flash EEPROM threshold instabilities due to charge trapping during program/erase cycling," *IEEE Trans. Device Mater. Rel.*, vol. 4, no. 3, pp. 335–344, Sep. 2004.

[36] C. A. Aslam, Y. L. Guan, and K. Cai, "Decision-directed retention-failure recovery with channel update for MLC NAND flash memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 65, no. 1, pp. 353–365, Jan. 2018.

[37] G. Dong, S. Li, and T. Zhang, "Using data postcompensation and predistortion to tolerate cell-to-cell interference in MLC NAND flash memory," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 10, pp. 2718–2728, Oct. 2010.

[38] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. NIPS*, vol. 27, 2014, pp. 1–9.

[39] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, 1988.

[40] A. Paszke et al., "Pytorch: An imperative style, high-performance deep learning library," in *Proc. NIPS*, vol. 32, 2019, pp. 1–12.

[41] X.-Y. Hu, E. Eleftheriou, and D. M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 386–398, Jan. 2005.

[42] J. Chen, A. Dholakia, E. Eleftheriou, M. P. C. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, Aug. 2005.

Zhen Mei (Member, IEEE) received the Ph.D. degree from Newcastle University, U.K., in 2017. He worked as a Post-Doctoral Researcher with the Singapore University of Technology and Design (SUTD) from 2017 to 2019 and as a System Engineer with Huawei Technologies Company Ltd., from 2019 to 2021. He is currently an Associate Professor with the Nanjing University of Science and Technology, China. His research interests include machine learning, information and coding theory for data storage, and communications systems.

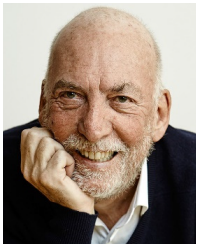
Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.

Kui Cai (Senior Member, IEEE) received the B.E. degree in information and control engineering from Shanghai Jiao Tong University, Shanghai, China, and the joint Ph.D. degree in electrical engineering from the Technical University of Eindhoven, The Netherlands, and the National University of Singapore. She is currently an Associate Professor with the Singapore University of Technology and Design (SUTD). Her main research interests include the areas of coding theory, information theory, and signal processing for various data storage systems and digital communications. She received the 2008 IEEE Communications Society Best Paper Award in Coding and Signal Processing for Data Storage. She was listed in the 2020 Who's Who in Engineering Singapore. She served as the Vice Chair (Academia) for the IEEE Communications Society, Data Storage.

Long Shi (Senior Member, IEEE) received the Ph.D. degree in electrical engineering from the University of New South Wales, Sydney, Australia, in 2012. From 2013 to 2016, he was a Post-Doctoral Fellow with the Institute of Network Coding, The Chinese University of Hong Kong, China. From 2014 to 2017, he was a Lecturer with the Nanjing University of Aeronautics and Astronautics, Nanjing, China. From 2017 to 2020, he was a Research Fellow with the Singapore University of Technology and Design. He is currently a Professor with the School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing. His research interests include wireless communications, blockchain networks, and federated learning.

Jun Li (Senior Member, IEEE) received the Ph.D. degree in electronic engineering from Shanghai Jiao Tong University, Shanghai, China, in 2009. From January 2009 to June 2009, he worked with the Department of Research and Innovation, Alcatel-Lucent Shanghai Bell, as a Research Scientist. From June 2009 to April 2012, he was a Post-Doctoral Fellow with the School of Electrical Engineering and Telecommunications, University of New South Wales, Australia. From April 2012 to June 2015, he was a Research Fellow with the School of Electrical Engineering, The University of Sydney, Australia. Since June 2015, he has been a Professor with the School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing, China. He was a Visiting Professor with Princeton University from 2018 to 2019. He has coauthored more than 200 papers in IEEE journals and conferences and holds one U.S. patent and more than ten Chinese patents in these areas. His research interests include network information theory, game theory, distributed intelligence, multiple agent reinforcement learning, and their applications in ultra-dense wireless networks, mobile edge computing, network privacy and security, and industrial Internet of Things. He serves as a TPC member for several flagship IEEE conferences. He is currently serving as an Editor for IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS.

Li Chen (Senior Member, IEEE) received the B.Sc. degree in applied physics from Jinan University, China, in 2003, and the M.Sc. degree in communications and signal processing and the Ph.D. degree in communications engineering from Newcastle University, U.K., in 2004 and 2008, respectively. From 2007 to 2010, he was a Research Associate with Newcastle University. In 2010, he returned to China as a Lecturer with the School of Information Science and Technology, Sun Yat-sen University, Guangzhou. From 2011 to 2012, he was a Visiting Researcher with the Institute of Network Coding, The Chinese University of Hong Kong. From 2011 to 2016, he was an Associate Professor and a Professor with The Chinese University of Hong Kong. Since 2013, he has been the Associate Head of the Department of Electronic and Communication Engineering (ECE). From July 2015 to October 2015, he was a Visitor with the Institute of Communications Engineering, Ulm University, Germany. From October 2015 to June 2016, he was a Visiting Associate Professor with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, IN, USA. From 2017 to 2020, he was the Deputy Dean of the School of Electronics and Communication Engineering. His research interests include information theory, error correction codes, and data communications. He is currently a Senior Member of the Chinese Institute of Electronics (CIE). He is a member of the IEEE Information Theory Society Board of Governors Conference Committee and chairing the Conference Committee. He was awarded The Chinese Information Theory Young Researcher Award by the Chinese Society of Electronics. He founded and chairs the IEEE Information Theory Society Guangzhou Chapter, which was awarded the 2021 Chapter of the Year of the IEEE Information Theory Society. He has been organizing several international conferences and workshops, including the 2018 IEEE Information Theory Workshop (ITW), Guangzhou, and the 2022 IEEE East Asian School of Information Theory (EASIT), Shenzhen, for which he is the General Co-Chair. He is also the TPC Co-Chair of 2022 IEEE/CIC International Conference on Communications in China (ICCC), Foshan. He will organize ISIT 2026, Guangzhou, China. He was an Associate Editor of IEEE TRANSACTIONS ON COMMUNICATIONS from 2018 to 2023. He is an Associate Editor of IEEE TRANSACTIONS ON COMMUNICATIONS.



Kees A. Schouhamer Immink (Life Fellow, IEEE) was an Adjunct Professor with the Institute for Experimental Mathematics, University of Duisburg-Essen, Germany, from 1994 to 2014. In 1998, he founded Turing Machines Inc., an innovative start-up focused on novel signal processing for DNA-based storage, where he currently holds the position of president. He designed coding techniques for digital audio and video recording products, such as Compact Disc, CD-ROM, DCC, DVD, Blu-ray Disc, and DCC. He received the Knighthood in 2000, the Personal Emmy Award in 2004, the 2017 IEEE Medal of Honor, the 1999 AES Gold Medal, the 2004 SMPTE Progress Medal, the 2014 Eduard Rhein Prize for Technology, the 2015 IET Faraday Medal, and the 1999 IEEE Edison Medal. He was elected into the Royal Netherlands Academy of Sciences, the Royal Holland Society of Sciences, and the (U.S.) National Academy of Engineering. He received the Honorary Doctorate from the University of Johannesburg in 2014.

Authorized licensed use limited to: SUN YAT-SEN UNIVERSITY. Downloaded on June 19,2024 at 06:44:30 UTC from IEEE Xplore. Restrictions apply.