Code-Aware Storage Channel Modeling via Machine Learning

Simeng Zheng and **Paul H. Siegel**
Electrical and Computer Engineering Dept., University of California, San Diego, La Jolla, CA 92093 U.S.A
{sizheng,psiegel}@ucsd.edu

2022 IEEE Information Theory Workshop (ITW) | 978-1-6654-8341-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITW54588.2022.9965920

***Abstract*—With the reduction in device size and the increase in cell bit-density, NAND flash memory suffers from larger inter-cell interference (ICI) and disturbance effects. Constrained coding can mitigate the ICI effects by avoiding problematic error-prone patterns, but designing powerful constrained codes requires a comprehensive understanding of the flash memory channel. Recently, we proposed a modeling approach using conditional generative networks to accurately capture the spatio-temporal characteristics of the read signals produced by arrays of flash memory cells under program/erase (P/E) cycling. In this paper, we introduce a novel machine learning framework for extending the generative modeling approach to the coded storage channel. To reduce the experimental overhead associated with collecting extensive measurements from constrained program/read data, we train the generative models via transferring knowledge from models pre-trained with pseudo-random data. This technique can accelerate the training process and improve model accuracy in reconstructing the read voltages induced by constrained input data throughout the flash memory lifetime. We analyze the quality of the model by comparing flash page bit error rates (BERs) derived from the generated and measured read voltage distributions. We envision that this machine learning framework will serve as a valuable tool in flash memory channel modeling to aid the design of stronger and more efficient coding schemes.**

***Index Terms*—Data storage systems, Machine learning, Constrained coding, Inter-cell interference.**

I. Introduction

Recently, there has been great interest in the application of machine learning in communications and networking, including data storage. For example, robust signal detection in magnetic recording channels using a recurrent neural network (RNN) architecture was demonstrated in [29]. A low-density parity-check (LDPC) decoder with flexible code lengths and column weights exploiting RNN was proposed in [26]. Machine learning was also applied to page failure prediction [14], [22] and, in a limited setting, read voltage generation [15] in NAND flash memory. The synergy between machine learning and data storage is stimulating important mutual progress.

Realistic models for storage and communication channels are critical tools in the design of signal processing and coding methods. Generative flash modeling (GFM) [28] was recently proposed to model the complex spatio-temporal characteristics of read voltages in flash memory channels. Although statistical models [7], [10], [16], [19] to characterize the effects of P/E cycling, ICI, and retention on flash memory read voltages have been proposed, their predictions have not been validated in the literature by comparing to measurements of pattern-dependent errors as a function of spatial and temporal factors.

GFM uses a conditional VAE-GAN [9] architecture, combining a variational auto-encoder (VAE) [8] and a generative adversarial network (GAN) [3] in a conditional setting. This modeling approach was shown to comprehensively learn both spatial and temporal properties of the flash channel.

Constrained codes [17] have been proposed to mitigate read errors arising from the ICI phenomenon in flash memory by forbidding the programming of error-prone patterns. Learning the characteristics of the input-constrained flash channel poses a challenge, however. Statistical models have not been used to explore the subtle characteristics of the channel associated with the use of constrained data. GFM has the potential to model the input-constrained channel, but a model trained from pseudo-random data does not provide sufficient knowledge about the constrained channel. On the other hand, acquiring a large dataset of measurements from constrained data can consume excessive amounts of time and hardware resources.

It has been observed that learned knowledge from models pre-trained on a large dataset (e.g., ImageNet [2]) can effectively be applied to other tasks, either by extracting off-the-shelf features from trained networks [21], [27], or by adapting learned knowledge to a new domain [18]. Moreover, transferring learning has shown success in the context of generative models, e.g., in applications to image generation [25]. To accurately model the input-constrained flash channel, we therefore propose a transfer learning approach, whereby the GFM network is first trained on a large dataset of measurements from pseudo-random data, and then is fine-tuned by re-training on a much smaller dataset of measurements from constrained data. We refer to this as code-aware GFM.

The paper has the following contributions:

1) We propose a novel framework for code-aware generative channel modeling, where the voltage levels of coded program levels can be precisely and rapidly reconstructed.
2) We show how generative models trained on pseudo-random programming data can efficiently transfer knowledge to other coded-channel modeling tasks where code-specific data is limited.
3) We demonstrate the quality of reconstruction in code-aware GFM by analysis of voltage distributions and bit error rates (BERs).

II. NAND Flash Memory and ICI Mitigation

A. NAND Flash Memory Basics

NAND flash memory stores data as voltages in floating gate transistors, called cells. In a flash chip, cells are organized in two-dimensional (2-D) arrays, called blocks, consisting

TABLE I
NUMERICAL VALUES OF PATTERN-DEPENDENT ERROR RATES FOR THE MOST SEVERE ICI PATTERNS

	Error rate	$[7\ 0\ 7]$	$[7\ 0\ 6]$	$[6\ 0\ 7]$	$\begin{bmatrix}7\\0\\7\end{bmatrix}$	$\begin{bmatrix}7\\0\\6\end{bmatrix}$	$\begin{bmatrix}6\\0\\7\end{bmatrix}$	$\begin{bmatrix}&7&\\7&0&7\\&7&\end{bmatrix}$	$\begin{bmatrix}&7&\\7&0&6\\&7&\end{bmatrix}$	$\begin{bmatrix}&7&\\7&0&7\\&6&\end{bmatrix}$
4000 P/E	2.45%	10.97%	7.42%	7.36%	15.42%	10.76%	8.78%	48.06%	35.64%	36.59%
7000 P/E	4.06%	14.45%	10.35%	10.15%	20.48%	15.01%	12.81%	50.34%	42.15%	37.39%
10000 P/E	5.84%	18.42%	13.31%	13.46%	25.73%	19.35%	17.11%	54.22%	44.42%	44.67%

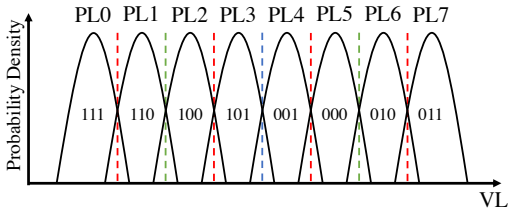

Fig. 1. Voltage distributions and a recursive alternate Gray mapping (RAGM) between cell program levels and binary logic values of a TLC NAND flash memory.

of horizontal wordlines (WLs) and vertical bitlines (BLs). Multilevel flash memories store multiple bits per cell. For example, a triple-level cell (TLC) memory stores three bits using 2^3=8 possible voltage levels. Within each block, the three bits stored in cells along a WL are logically grouped into three pages, called the lower, middle, and upper page, respectively.

There are three basic operations on a flash device: program (write), read, and erase. We denote the program level as PL and the read voltage level as VL. Fig. 1 illustrates the the conditional probability density functions (PDFs) of read voltages for 8 program levels, each corresponding to a 3-bit string of lower, middle, and upper bits. The dash-dotted vertical lines represent the read thresholds used to recover the stored data. Level errors and bit errors occur when, for example, PL=0 induces a read voltage VL lying above the first threshold and below the second threshold, causing the level to be mistakenly detected as 1 and the the upper bit to be mistakenly detected as 0.

B. ICI Mitigation via Constrained Coding

ICI effects, caused by parasitic capacitive coupling between flash cells, is one of the major obstacles to accurate programming and reading of a flash device [1]. Severe ICI arises when three consecutive cells in WL or BL directions are programmed to high-low-high levels.

Error rates for the most severe ICI patterns in a commercial TLC flash device are shown in Table I. Using (i, j) to denote the (WL,BL) position of a cell in the block and $V_{th(01)}$ to denote the threshold between PL=0 and PL=1, the table gives the overall level error rate for PL=0, and the error rates for worst-case WL, BL, and 2-D patterns, or, mathematically,

$$P(\mathrm{VL}_{(i,j)} > V_{th(01)} | \mathrm{PL}_{(i,j)}{=}0);$$
$$P(\mathrm{VL}_{(i,j)} > V_{th(01)} | \mathrm{PL}_{(i,j-1)}, \mathrm{PL}_{(i,j)}{=}0, \mathrm{PL}_{(i,j+1)});$$
$$P(\mathrm{VL}_{(i,j)} > V_{th(01)} | \mathrm{PL}_{(i-1,j)}, \mathrm{PL}_{(i,j)}{=}0, \mathrm{PL}_{(i+1,j)});$$
$$P(\mathrm{VL}_{(i,j)} > V_{th(01)} | \mathrm{PL}_{(i\pm1,j)}, \mathrm{PL}_{(i,j)}{=}0, \mathrm{PL}_{(i,j\pm1)}).$$

We make two observations from Table I. First, ICI significantly increases error rates. At 4000 P/E cycles, the error rate of 707 pattern in WL (resp., BL) direction is a factor of 4.5 (resp., 6.3) larger than the average error rate. If we program 707 in both directions, the error rate is a factor of 19.6 larger than the average error rate. Second, P/E cycling causes error rates to increase. Specifically, the average error rate increases by a factor of 2.38 from 4000 P/E cycles to 10000 P/E cycles. For dominant 707 error patterns in WLs (resp., BLs), the error rate increases by a factor of 1.68 (resp., 1.67) from 4000 P/E cycles to 10000 P/E cycles.

Solid-state drives (SSDs) employ powerful error-correction codes (ECCs) [6] within their controllers to cope with such errors. Constrained codes to further reduce ICI-induced errors have been proposed and some have been experimentally validated [4], [5], [20], [24]. In particular, read-and-run (RR) constrained coding techniques [5] efficiently eliminate selected detrimental patterns by coding on only one page per WL. They allow random page access and are compatible with page-based ECCs. A generative model that accurately learns input-constrained channels will be a valuable tool in optimizing the combination of constrained coding and ECC.

III. CODE-AWARE STORAGE CHANNEL MODELING

The GFM scheme in [28] learns an approximation to the intractable likelihood $P(\mathrm{VL}|\mathrm{PL}, \mathrm{P/E})$ from a dataset of measured voltage arrays VL produced by pseudo-random (unconstrained) program arrays PL. The goal of code-aware channel modeling is to infer the intractable likelihood $P(\mathrm{VL}^{\mathcal{S}}|\mathrm{PL}^{\mathcal{S}}, \mathrm{P/E})$, where $\mathrm{VL}^{\mathcal{S}}$ is the voltage array produced by the code-constrained program array $\mathrm{PL}^{\mathcal{S}}$. In this section, we describe our transfer learning approach to achieving this goal.

A. Review of Generative Flash Modeling

The conditional VAE-GAN architecture underlying the GFM scheme consists of three modules: encoder (Enc), generator (Gen), and discriminator (Dis).

During the training process, the encoder Enc produces latent vectors z from VL based on the VAE technique [8]. The generator Gen reconstructs an array of read voltages, $\widetilde{\mathrm{VL}}$, based on PL, P/E, and z. The P/E vectors are concatenated with the output features of Gen for spatio-temporal combination. The discriminator Dis is trained to distinguish real VL from fake VL.

After optimization, the learned Gen serves as a realistic flash channel simulator which accepts program level array PL, P/E cycle count, and latent vector z as inputs. The latent vector is sampled from a standard multivariate Gaussian distribution. We express the reconstruction of VL in the training and evaluation processes, respectively, as

$$\text{(Train) } \widetilde{\mathrm{VL}} = Gen(\mathrm{PL}, \mathrm{P/E}, Enc(\mathrm{VL}))$$
$$\text{(Evaluation) } \widetilde{\mathrm{VL}} = Gen(\mathrm{PL}, \mathrm{P/E}, z).$$

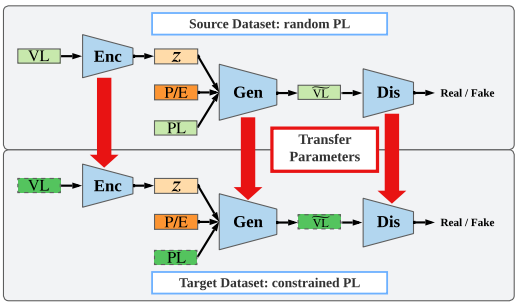

Fig. 2. Pipeline of code-aware generative flash modeling.

Full details about the training, evaluation, and experimental results are given in [28].

The GFM approach was demonstrated to accurately reconstruct cell voltage levels by capturing spatial ICI effects and temporal distortions from P/E cycling, as validated by comparing predicted time-dependent and pattern-dependent errors to error measurements.

B. Code-aware Generative Flash Modeling

The GFM framework is capable of learning the likelihood $P(\mathrm{VL}^{\mathcal{S}}|\mathrm{PL}^{\mathcal{S}}, \mathrm{P/E})$ from a sufficiently large dataset $\{(\mathrm{PL}^{\mathcal{S}}, \mathrm{VL}^{\mathcal{S}}, \mathrm{P/E})\}$ of code-constrained programming measurements at each P/E cycle. To avoid the expense of producing such a large dataset, we propose to use a transfer learning approach. We pre-train the GFM network on a large-scale source dataset $\{(\mathrm{PL}, \mathrm{VL}, \mathrm{P/E})\}$ of VL measurements from pseudo-random (unconstrained) program arrays PL, then fine-tune it using a much smaller target dataset $\{(\mathrm{PL}^{\mathcal{S}}, \mathrm{VL}^{\mathcal{S}}, \mathrm{P/E})\}$ of code-constrained measurements.

We now formulate the pipeline of code-aware GFM. As shown in Fig. 2, at the beginning of training, three network modules in GFM (Enc, Gen, and Dis) are initialized with pre-trained weights learned from source dataset. Using the target dataset, the code-aware GFM follows the framework of GFM to finish the training process. After training, the network parameters in Gen represent the simulator to produce voltage levels from code-constrained PL arrays.

We note that the relation between source and target datasets can impact the transfer learning results. In our case, because random programming arrays very likely include constrained sub-arrays, sharing the pre-trained network weights during the fine-tuning step enables the transfer of relevant knowledge.

C. Transfer Learning Configuration

It has been observed that pre-training for all network modules can provide better results than pre-training for one individual module [25]. Therefore, in our transfer learning configuration, we share the parameters of all three modules of the pre-trained network.

In our experiments, we consider two read-and-run (RR) constrained codes [5]. The corresponding target datasets $\{(\mathrm{PL}^{\mathcal{S}}, \mathrm{VL}^{\mathcal{S}}, \mathrm{P/E})\}$ consist of pairs of 64×64 PL and VL arrays, collected from a commercial TLC flash device at selected P/E cycles, as in the original GFM setup in [28].

This framework is also applicable to data shaping codes for flash memory [11]–[13]. These codes minimize the average cell wear due to programming by optimally "shaping" the probability distribution of the programmed cell levels.

TABLE II
SIZES OF TRAINING AND EVALUATION DATASETS

	Training	Evaluation
$\lvert\{(\mathrm{PL}, \mathrm{VL}, \mathrm{P/E})\}\rvert$	1.5×10^5	2.1×10^4
$\lvert\{(\mathrm{PL}^{\mathcal{S}_{WL}}, \mathrm{VL}^{\mathcal{S}_{WL}}, \mathrm{P/E})\}\rvert$	1.5×10^4	1.5×10^4
$\lvert\{(\mathrm{PL}^{\mathcal{S}_{2D}}, \mathrm{VL}^{\mathcal{S}_{2D}}, \mathrm{P/E})\}\rvert$	1.5×10^4	1.5×10^4

The two constrained datasets are collected from a single commercial 1X-nm TLC chip belonging to the same family of chips used for the GFM experiments in [28]. Due to the variation of mappings between manufacturers and product generations, we describe the disallowed patterns of the code-constrained data in terms of the mapping in Fig. 1. The first target dataset uses a code constraint $\mathcal{S}_{WL}$ that forbids $\{000, 010\}$ in the lower page of each WL. This eliminates error-prone patterns containing 707, 706, and 607 in the WL direction, as well as other high-low-high error-prone patterns.

The second target dataset uses a code constraint $\mathcal{S}_{2D}$ that forbids $\{000, 010\}$ in lower bits along both WL and BL directions. This eliminates error-prone patterns containing 707, 706, and 607 in both WLs and BLs, including all patterns shown in Table I, as well as other patterns.

We implement the WL-based constraint $\mathcal{S}_{WL}$ with an interleaved, rate 12:18 run-length limited (RLL) $(d, k) = (0, 1)$ code of overall block length 36 on the lower page, yielding an effective rate of 0.89 [5], [23]. The 2D-constraint is implemented with the 2D RR scheme in [5], [23], which has an effective rate of 0.83.

We collect equal numbers of measured voltages at three P/E cycle counts: 4000, 7000, and 10000. The training and evaluation dataset sizes used in our modeling experiments, described in the next section, are shown in Table II. Note that the size of the target datasets is only 10% of the size of the source dataset.

Remark 1. *In all transfer learning experiments, we use the same settings as were used to train the GFM, namely, batch size 2 and learning rate 2×10^{-4}. We settled upon these training parameters after several experiments.*

IV. EXPERIMENTAL RESULTS AND ANALYSIS

In this section, we evaluate our code-aware GFM framework and present results of its application to the two RR constrained codes described in the previous section. We use two evaluation criteria, one to measure the accuracy of the reconstructed results, and the other to measure the training efficiency of the transfer learning procedure. The former is based on probability density functions (PDFs) of the reconstructed voltages, and the latter is based on the number of training iterations required to achieve accurate reconstruction. The evaluation metrics are defined in more detail below.

1) Probability density functions (PDFs): The read voltage PDFs are useful in optimizing read thresholds, gauging cell wear, and estimating bit error rates (BERs). For each P/E cycle, we estimate the conditional PDFs by the frequency of occurrence of measured voltage levels for each given program level. In addition to visually comparing the measured PDFs and reconstructed PDFs,

TABLE III
MODELING EXPERIMENTS AND TRAINING ITERATIONS

Initialization (I)	Training (T)	Evaluation (E)	Training Iterations
Random	PR	PR,WL,2D	5.25×10^5
	WL	WL	6×10^4
	2D	2D	6.75×10^4
Pre-trained	WL	WL	$\mathbf{7.5 \times 10^3}$
	2D	2D	$\mathbf{7.5 \times 10^3}$

we compute the total variation distance between the two PDFs and compare the associated bit error rates (BERs) on the lower, middle, and upper pages.

2) Training iterations: The number of training iterations needed to achieve satisfactory results can be used as a metric to evaluate the "speed" of the transfer learning process. A training iteration is defined as a single update of the model weights during training. For example, in [28], training the GFM network takes 7 epochs with a batch size of 2 using random programming arrays. With the training dataset size in Table II, the total number of training iterations is 5.25×10^5.

A. Experimental Settings

We conducted a matrix of experiments to evaluate the effectiveness of transfer learning in code-aware GFM, as summarized in Table III. (See discussion below for an explanation of the abbreviations in the table.) The training iterations are shown in the last column of the table. For convenience, we use a shorthand notation to distinguish the experiments according to the training dataset ("T"), the network initialization ("I"), and the evaluation dataset ("E").

The training dataset corresponded to program arrays based on either pseudo-random data ("T-PR"), $\mathcal{S}_{WL}$-constrained data ("T-WL"), or $\mathcal{S}_{2D}$-constrained data ("T-2D"). Regarding the training mode, training started either from randomly initialized network weights ("I-Rnd") or pre-trained weights ("I-Pre") from T-PR training.

The evaluation mode examined reconstructed voltages generated by pseudo-random data ("E-PR"), $\mathcal{S}_{WL}$-constrained data ("E-WL"), or $\mathcal{S}_{2D}$-constrained data ("E-2D"). Comparisons are made to measurements ("M") from the TLC chip, derived from the pseudo-random dataset ("M-PR"), the $\mathcal{S}_{WL}$-constrained dataset ("M-WL"), or the $\mathcal{S}_{2D}$-constrained dataset ("M-2D").

We present results for the following experiments,

1) **M-PR, M-WL, M-2D**: These represent baseline experimental measurements from several 1X-nm flash blocks programmed with pseudo-random, WL-constrained, or 2D-constrained data.
2) **I-Rnd / T-PR / E-(PR,WL,2D)**: We train GFM with random initial network weights using the pseudo-random training dataset and evaluate with pseudo-random data, WL-constrained data, and 2D-constrained data.
3) **I-Pre / T-WL / E-WL**: We initialize GFM with pre-trained weights from the previous training experiment (I-Rnd/T-PR), fine-tune the network using WL-constrained data, and evaluate the model with WL-constrained data.
4) **I-Rnd / T-WL / E-WL**: We train GFM with random initial network weights using the WL-constrained training dataset and evaluate with WL-constrained data.

TABLE IV
TOTAL VARIATION DISTANCE

P/E Cycle Count	4000	7000	10000
$d_{TV}(P_{\text{M-PR}}, P_{\text{I-Rnd/T-PR/E-PR}})$	0.0688	0.0650	0.0687
$d_{TV}(P_{\text{M-WL}}, P_{\text{I-Pre/T-WL/E-WL}})$	0.0696	0.0535	0.0505
$d_{TV}(P_{\text{M-WL}}, P_{\text{I-Rnd/T-WL/E-WL}})$	0.1421	0.1300	0.1020
$d_{TV}(P_{\text{M-WL}}, P_{\text{I-Rnd/T-PR/E-WL}})$	0.1068	0.1116	0.1181
$d_{TV}(P_{\text{M-2D}}, P_{\text{I-Pre/T-2D/E-2D}})$	0.1007	0.0771	0.0908
$d_{TV}(P_{\text{M-2D}}, P_{\text{I-Rnd/T-2D/E-2D}})$	0.1175	0.1021	0.1408
$d_{TV}(P_{\text{M-2D}}, P_{\text{I-Rnd/T-PR/E-2D}})$	0.1470	0.1330	0.1364

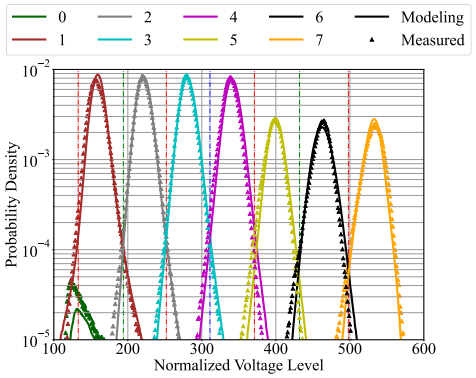

Fig. 3. PDF plots in logarithmic scale for measured and regenerated voltage levels (experiment I-Pre/T-WL/E-WL) at 7000 P/E cycles. The visualization is based on dataset $\{(\text{PL}^{\mathcal{S}_{WL}}, \text{VL}^{\mathcal{S}_{WL}}, \text{P/E})\}$.

5) **I-Pre / T-2D / E-2D**: We initialize GFM with pre-trained weights from the first training experiment (I-Rnd/T-PR), fine-tune the network using the 2D-constrained dataset, and evaluate the model with 2D-constrained data.
6) **I-Rnd / T-2D / E-2D**: We train GFM with random initial network weights using the 2D-constrained training dataset and evaluate with 2D-constrained data.

B. PDF Analysis

We now qualitatively and quantitatively analyze the reconstructed voltages from code-aware GFM. First, we visualize the PDFs of the measured and reconstructed read voltages. Fig. 3 shows the normalized conditional PDFs of the eight TLC program levels in the reconstructed data for experiment I-Pre/T-WL/E-WL at 7000 P/E cycles. (The plots of voltage PDFs for this experiment at 4000 and 10000 P/E cycles yield qualitatively similar results.) In this log-linear plot, the y-axis represents the probability density and the x-axis represents the read voltages using an arbitrary scale.

Note that the $\mathcal{S}_{WL}$ code constraint on lower pages induces a smaller probability of occurrence for PLs $5, 6, 7$, which is approximately $\frac{1}{3}$ of that of PLs $1, 2, 3, 4$. Qualitatively, the PDFs generated by code-aware GFM (solid curves) closely match the measured PDFs (triangle markers). Similarly, in experiment I-Pre/T-2D/E-2D, the visualization of the model-generated PDFs accurately reflects the measured PDFs and their dependence on P/E cycles.

Next, we evaluate the PDF results of the code-aware GFM experiments quantitatively using total variation (TV) distance, d_{TV}. This distance provides a measure of the difference between the real (measured) distributions P_{real} and the fake (reconstructed) distributions P_{fake},

$$d_{TV}(P_{real}, P_{fake}) = \frac{1}{2} \sum_{\text{VL}} |P_{real}(\text{VL}) - P_{fake}(\text{VL})|.$$

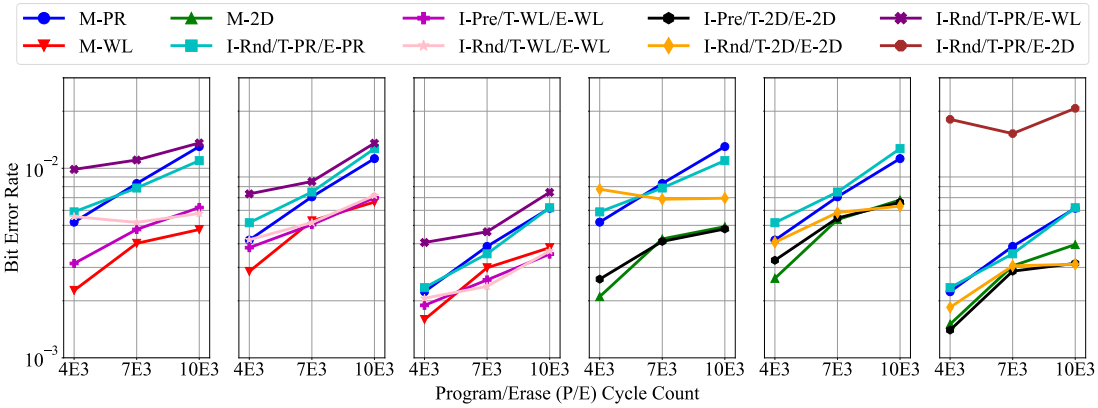

Fig. 4. BER comparisons: the leftmost (resp., rightmost) three sub-figures show lower, middle, and upper page BERs for $\mathcal{S}_{WL}$-coded (resp., $\mathcal{S}_{2D}$-coded) data.

The numerical results are shown in Table IV. We find that pre-training helps code-aware GFM produce distributions with the least TV distance in both $\mathcal{S}_{WL}$-coded and $\mathcal{S}_{2D}$-coded scenarios.

It is also important to consider the tails of the distributions, which have a major impact on the channel error rate. As discussed in Section II-A, cell level errors are determined by comparing the read voltages to the read thresholds, and the resulting bit errors on pages arise from the mapping between cell levels and their corresponding 3-bit binary logic values. We compared measured and reconstructed page bit error rates (BERs) from the ten experiments described in Section IV-A. The results are shown in Fig. 4.

The leftmost three sub-figures in Fig. 4 pertain to the lower, middle, and upper pages in the $\mathcal{S}_{WL}$-coded case, respectively. The six curves in each plot correspond to the experimental measurements M-PR and M-WL, the GFM modeling experiments I-Rnd/T-PR/E-PR and I-Rnd/T-PR/E-WL, and the code-aware GFM experiments I-Rnd/T-WL/E-WL and I-Pre/T-WL/E-WL using the $\mathcal{S}_{WL}$ dataset, comparing training "from scratch" and with pre-trained network parameters.

The M-PR and M-WL curves show that $\mathcal{S}_{WL}$ coding decreases the measured BER on all three pages at all three measured P/E cycles, confirming the observations in [5]. The GFM experiment I-Rnd/T-PR/E-PR using random initialization along with training and evaluation on pseudo-random data reconstructs page BERs quite accurately at all three P/E cycles, a finding that is consistent with [28]. However, when this GFM network is evaluated using the $\mathcal{S}_{WL}$-coded dataset in experiment I-Rnd/T-PR/E-WL, we see that the reconstructed BERs are significantly higher than the measured BERs in the M-WL curve at all three P/E cycles. This suggests that the pseudo-random dataset does not sufficiently capture all of the characteristics of the coded channel.

The final two curves compare the effects of random initialization and pre-training in the code-aware GFM networks obtained by training and evaluating on the $\mathcal{S}_{WL}$ dataset. We see that the two experiments yield very similar reconstructed BERs, with the exception of the lower page BER at 4000 P/E cycles, where random initialization yields a noticeably more inaccurate estimate. Overall, the reconstructed BERs qualitatively track the measured M-WL results reasonably well, although both models overestimate BER in lower pages at all P/E cycles, as well as in middle pages at 4000 P/E cycles.

The rightmost three sub-figures in Fig. 4 show the corresponding BER results for lower, middle, and upper pages in the $\mathcal{S}_{SD}$-coded case, respectively. (The BERs of I-Rnd/T-PR/E-2D for lower and middle pages are at least 3×10^{-2}; thus, the curves are not shown in the sub-figures.) The overall conclusions drawn from these curves are similar to the $\mathcal{S}_{WL}$-coded case, although we see that the GFM trained on the pseudo-random source dataset does an even worse job of learning the $\mathcal{S}_{2D}$-coded channel.

C. Iteration Number Analysis

The number of training iterations used in the experiments was determined by comparing the reconstructed PDFs to the corresponding measured PDFs using TV distance.

From Table III, we find that the number of iterations required to fine-tune the code-aware GFM network from the pre-trained model, 7.5×10^3, is only 12.5% (resp., 11.11%) of the number required when training from scratch using the target $\mathcal{S}_{WL}$ (resp., $\mathcal{S}_{SD}$) dataset, namely 6×10^4 (resp., 6.75×10^4).

Specifically, when training from scratch using the smaller target dataset, we observed that in the early training iterations the reconstructed read voltage PDFs do not accurately capture temporal P/E cycle variations and tail behavior. On the other hand, adaptation from a single GFM network pre-trained with a sufficiently large source dataset of pseudo-random data provides enough channel knowledge to significantly accelerate the learning process from both of the smaller target datasets.

V. Conclusion

This paper presents an application of transfer learning to generative modeling of read voltages in flash memory channels. We fine-tune a generative model pre-trained with a large source dataset of pseudo-random spatio-temporal data using much smaller code-constrained target datasets. By comparing measured and reconstructed read voltage probability distribution functions and page bit error rates in a commercial TLC flash memory, we demonstrate that pre-training can accelerate learning for multiple generative modeling tasks even when the amount of target training data is very limited. These results motivate further investigation into the use of transfer learning in applications of machine learning to data storage and communication systems.

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. of the IEEE*, vol. 105, no. 9, pp. 1666–1704, Sep. 2017.

[2] J. Deng, W. Dong, R. Socher, L-J. Li, K. Li, and F-F. Li, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Miami, FL, USA, June, 2009.

[3] I. J. Goodfellow, J. P.-Abadie, M. Mirza, B. Xu, D. W.-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Montréal, Canada, Dec. 2014, pp. 2672–2680.

[4] A. Hareedy, B. Dabak, and R. Calderbank, "Managing device lifecycle: reconfigurable constrained codes for M/T/Q/P-LC flash memories," *IEEE Trans. Inf. Theory*, vol. 67, no. 1, pp. 282–295, Oct. 2020.

[5] A. Hareedy, S. Zheng, P. H. Siegel, and R. Calderbank, "Read-and-run constrained coding for modern flash devices," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Seoul, South Korea, May 2022.

[6] P. Huang, Y. Liu, X. Zhang, P. H. Siegel, E. F. Haratsch, "Syndrome-coupled rate-compatible error-correcting codes: theory and application," *IEEE Trans. Inf. Theory*, vol. 66, no. 4, pp. 2311–2330, Jan. 2020.

[7] Y. Kim and B. V. K. Vijaya Kumar, "Writing on dirty flash memory," in *Proc. 52nd Annu. Allerton Conf. Commun., Control, Comput.*, Monticello, IL, USA, Oct. 2014, pp. 513–520.

[8] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Represent. Learn. (ICLR)*, Banff, Canada, Apr. 2014.

[9] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, June 2016.

[10] Q. Li, A. Jiang, and E. F. Haratsch, "Noise modeling and capacity analysis for NAND flash memories," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Honolulu, HI, USA, June. 2014, pp. 2262–2266.

[11] Y. Liu, P. Huang, A. W. Bergman, and P. H. Siegel, "Rate-constrained shaping codes for structured sources," *IEEE Trans. Inf. Theory*, vol. 66, no. 8, pp. 5261–5281, Aug. 2020.

[12] Y. Liu, Y. Li, P. Huang, and P. H. Siegel, "Rate-constrained shaping codes for finite-state channels with cost," in *Proc. IEEE Int. Symp. Inf. Theory*, Espoo, Finland, Jun. 26-Jul. 1, 2022, to be published.

[13] Y. Liu and P. H. Siegel, "Shaping codes for structured data," in *Proc. IEEE Global Commun. Conf.*, Washington, DC, USA, Dec. 2016.

[14] Y. Liu, S. Wu, and P. H. Siegel, "Bad Page Detector for NAND Flash Memory," in *Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, USA, Mar. 2020.

[15] Z. Liu, Y. Liu, and P. H. Siegel, "Generative modeling of NAND flash memory voltage level," in *Non-Volatile Memories Workshop (NVMW)*, La Jolla, CA, USA, Mar. 2021.

[16] 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. Select. Areas Commun.*, vol. 34, no. 9, pp. 2294-2311, Sept. 2016.

[17] B. H. Marcus, R. M. Roth, and P. H. Siegel. (Oct. 2001). *An Introduction to Coding for Constrained Systems.* [Online]. Available: https://personal.math.ubc.ca/ marcus/Handbook/.

[18] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Columbus, OH, USA, June, 2014.

[19] T. Parnell, N. Papandreou, T. Mittelholzer, and H. Pozidis, "Modelling of the threshold voltage distributions of sub-20nm NAND flash memory," in *Proc. IEEE Global Commun. Conf.*, Austin, TX, USA, Dec. 2014, pp. 2351–2356.

[20] M. Qin, E. Yaakobi, and P. H. Siegel, "Constrained codes that mitigate inter-cell interference in read/write cycles for flash memories," *IEEE J. Select. Areas Commun.*, vol.32, no.5, pp. 836–846, May 2014.

[21] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshop*, Columbus, OH, USA, June 2014.

[22] N. Sree Prem, "An Application of Machine Learning to Bad Page Prediction in Multilevel Flash," Master's Thesis, University of California San Diego, 2019.

[23] P. H. Siegel, "Constrained Codes for Multilevel Flash Memory," presented at *North American School of Information Theory (Padovani Lecture)*, La Jolla, California, Aug. 12, 2015. Available: http://cmrr-star.ucsd.edu/static/presentations/Padovani_Lecture_NASIT_Website.pdf. Video: https://www.youtube.com/watch?v=FCv2PJryUr4.

[24] V. Taranalli, H. Uchikawa, and P. H. Siegel, "Error analysis and inter-cell interference mitigation in multi-level cell flash memories," in *Proc. IEEE Int. Conf. Commun. (ICC)*, London, U.K., June 2015, pp. 271–276.

[25] Y. Wang, C. Wu, L. Herranz, J. van de Weijer, A. Gonzalez-Garcia, and B. Raducanu, "Transferring GANs: generating images from limited data," in *Proc. European Conf. Comput. Vis. (ECCV)*, Munich, Germany, Sep. 2018.

[26] X. Xiao, B. Vasić, R. Tandon, and S. Lin, "Designing finite alphabet iterative decoders of LDPC codes via recurrent quantized neural networks," *IEEE Trans. Commun.*, vol. 68, no. 7, pp. 3963–3974, Apr. 2020.

[27] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferrable are features in deep neural networks?" in *Proc. Neural Inf. Process. Syst. (NIPS)*, Montréal, Canada, Dec. 2014.

[28] S. Zheng, C-H. Ho, W. Peng, P. H. Siegel, "Spatio-temporal modeling for flash memory channels using conditional generative nets," May 2022, *arXiv*: 2111.10039.

[29] S. Zheng, Y. Liu, and P. H. Siegel, "PR-NN: RNN-based detection for coded partial-response channels," *IEEE J. Select. Areas Commun.*, vol. 39, no. 7, pp. 1967–1982, July 2021.