Bad Page Detector for NAND Flash Memory

Yi Liu, Si Wu, and Paul H. Siegel
Electrical and Computer Engineering Dept., University of California, San Diego, La Jolla, CA 92093 U.S.A
{yil333, siw070, psiegel}@ucsd.edu

Abstract—NAND flash memory cells gradually wear out during the program/erase (P/E) cycling, resulting in an increasing bit error count (BEC) at the page level. The BEC behavior varies significantly among pages. To increase the lifetime of a flash memory device, we propose a bad page detector, which predicts whether a page will become a “bad” page in the near future based on its current and previous BEC information. Two machine learning algorithms, based upon time-dependent neural network [1] and long-short term memory [2] architectures, were used to design the detector. Experimental results based on data collected from a TLC flash memory test platform quantify its effectiveness.

I. INTRODUCTION

NAND flash memory cells are able to endure only a limited number of write operations. The program/erase process causes a deterioration of the oxide layer that traps electrons in the cell and eventually makes the cell unreliable. This causes the bit error count (BEC) of a page to increase. If the BEC exceeds a given threshold, for example the correction capability of the error correction code (ECC), some of the data stored in the page is lost, and we say the page is a “bad” page. To address this problem, we develop a bad page detector. It predicts the behavior of a page in the near future based on the current and previous BEC information. If a potentially bad page is detected, it can be retired from use and no future data will be written to it. The idea of bad page detection was mentioned in [3], where a detector based on a support-vector machine (SVM) was considered. In this paper, we extend that work and propose two algorithms based on machine learning, including a time-dependent neural network (TDNN) and a long-short term memory (LSTM).

II. TRAINING DATA SET

We measure the performance of TLC flash memory by performing a program and erase (P/E) cycling experiment. The TLC flash memory blocks are P/E cycled up to 10,000 P/E cycles and the BECs are recorded every 100 P/E cycles. The data collected consists of sequences of page BECs, \( N(T) = [N(T_1), N(T_2), \ldots, N(T_u)] \), associated with P/E cycle counts \( T_1, T_2, \ldots, T_u \).

We now formally define a \((T, T_{Of}, N_{Th})\)-bad page.

**Definition 1.** In a flash memory device, a physical page is called a \((T, T_{Of}, N_{Th})\)-bad page if its BEC at some P/E cycle count less than \( T + T_{Of} \) is equal to or greater than \( N_{Th} \). Here \( T_{Of} \) is the P/E cycle count offset, and \( N_{Th} \) is the BEC threshold.

For example, the BEC of a \((3000, 500, 1200)\)-bad page surpasses 1200 at some P/E cycle count before 3500. Notice that this page is also a \((500, 1200)\)-bad page for any \( T \geq 3000 \).

Based on this definition, we introduce a labeling function, which takes \( T, T_{Of}, N_{Th} \) and a page’s BEC sequence \( N(T) \) as input. It is defined as

\[
L(T, T_{Of}, N_{Th}) = \begin{cases} 
1, & N(T') \geq N_{Th}, T' \leq T + T_{Of} \\
0, & \text{otherwise}.
\end{cases}
\]

(1)

The label 1 indicates that a page is a \((T, T_{Of}, N_{Th})\)-bad page.

The detector predicts \((T, 500, N_{Th})\)-bad pages based on BEC information before \( T \). The input of this detector is the BECs vector \([N(T_1), N(T_2), \ldots, N(T_5)]\) where \( T_i = T_{i-1} + 100 \) and \( T_5 = T \). The training dataset \( \mathcal{X} \) consists of pairs \((x, t)\), where \( x \) is a length-5 BEC vector,

\[
x = [N(T_1), N(T_2), N(T_3), N(T_4), N(T_5)],
\]

and \( t \) is defined by resizing \( T_5 \) according to

\[
t = \frac{T_5 - 4000}{10000 - 4000}
\]

such that \( t \) lies in the range \([0, 1]\). The P/E cycling experiment collected data from 12,855 pages, so the dataset size is 12,855 \times 52 = 668,460 (covering P/E cycle count windows 4000 – 4400 to 9100 – 9500). The BEC vector length 5 and P/E cycle count offset 500 were chosen for illustrative purposes.

III. TIME-DEPENDENT NEURAL NETWORK

In [3], an SVM detector was trained on a subset of \( \mathcal{X} \) with fixed \( t \). In this section, we discuss the structure of a time-dependent neural network (TDNN), whose weights and biases are functions of \( t \). Its motivation comes from the fact that BEC behavior varies as a function of P/E cycle count. The detector consists of three kinds of layers: time-dependent dense layer \( D(x, t) \) (TDDL), leaky ReLU layer \( L(x) \) and softmax layer \( S(x) \). Given an input vector \( x \in \mathbb{R}^n \), TDDL defines following operation

\[
y = D(x, t) = W(t)x + B(t)
\]

(4)

where \( y \in \mathbb{R}^n \) is the output vector and \( W \in \mathbb{R}^{n \times m}, B \in \mathbb{R}^n \) are network parameters to be determined during the training process. If the parameters are fixed, the layer is a dense layer, denoted simply by \( D(x) \).

The TDNN \( \Phi(x, t) \) contains two TDDLs \( D_0(x, t) \) and \( D_1(x, t) \) with input size 5 and output size 5; one dense layer \( D_2(x) \) with input size 5 and output size 2; two leaky ReLU layers \( L(x) \); and one softmax layer \( S(x) \). We build the network \( \Phi(x, t) \) as follows:

\[
y = \Phi(x, t) = S(D_2(L(D_1(L(D_0(x, t), t))))).
\]

(5)
The LSTM detector predicts bad pages by solving a regression problem. The network takes a sequence of $x_t$ in a TDDL can then be represented as

$$ W(t) = \{ w_{ij}(t) \} = \{ a_{ij}t^3 + b_{ij}t^2 + c_{ij}t + d_{ij} \} $$

where $a_{ij}, b_{ij}, c_{ij}, d_{ij}$ are control parameters that are determined during the training process. In general, higher degree polynomials would result in a more accurate approximation. The weight function $W(t)$ in a TDDL can then be represented as

$$ W(t) = \{ w_{ij}(t) \} = \{ a_{ij}t^3 + b_{ij}t^2 + c_{ij}t + d_{ij} \} $$

$$ = \{ a_{ij} \} t^3 + \{ b_{ij} \} t^2 + \{ c_{ij} \} t + \{ d_{ij} \} $$

$$ = W_a t^3 + W_b t^2 + W_c t + W_d. $$

A dense layer $D(x, t)$ can be converted to the form

$$ D(x, t) = D_a(x) t^3 + D_b(x) t^2 + D_c(x) t + D_d(x). $$

Therefore, a TDDL can be realized by connecting 4 dense layers in parallel, as shown in Fig. 2.

**IV. LONG-SHORT TERM MEMORY**

The LSTM detector predicts bad pages by solving a regression problem. The network takes a sequence of $l$–consecutive BECs and predicts the BEC at a future P/E cycle count. The labeling function $L_{LSTM}$ is given by

$$ L_{LSTM}(x) = N(T_5 + 500) $$

for any $x \in X$. The LSTM network consists of a standard LSTM layer $L : \mathbb{R}^4 \to \mathbb{R}^4$ and a dense layer $D : \mathbb{R}^4 \to \mathbb{R}$, represented as $y = D(M(x))$. For a specified $T_{\text{th}}$ (usually $N_{\text{th}}$), if $y \geq y_{\text{th}}$, we predict the page is a bad page.

**Fig. 2: The structure of time-dependent dense layer. DLs represent dense layers.**

Output $y$ is a length-2 vector $[p_1, p_2]$ that represents the probability of being a bad page ($p_1 + p_2 = 1$). We predict the page is a bad page if, for a specified threshold probability $p_{\text{th}}$ (usually set to 0.5), $p_1 \geq p_{\text{th}}$. The loss function used in the training process is cross entropy. In general, higher degree polynomials would result in a more accurate approximation. The weight function $W(t)$ in a TDDL can then be represented as

$$ W(t) = \{ w_{ij}(t) \} = \{ a_{ij}t^3 + b_{ij}t^2 + c_{ij}t + d_{ij} \} $$

$$ = \{ a_{ij} \} t^3 + \{ b_{ij} \} t^2 + \{ c_{ij} \} t + \{ d_{ij} \} $$

$$ = W_a t^3 + W_b t^2 + W_c t + W_d. $$

A dense layer $D(x, t)$ can be converted to the form

$$ D(x, t) = D_a(x) t^3 + D_b(x) t^2 + D_c(x) t + D_d(x). $$

Therefore, a TDDL can be realized by connecting 4 dense layers in parallel, as shown in Fig. 2.

**IV. LONG-SHORT TERM MEMORY**

The LSTM detector predicts bad pages by solving a regression problem. The network takes a sequence of $l$–consecutive BECs and predicts the BEC at a future P/E cycle count. The labeling function $L_{LSTM}$ is given by

$$ L_{LSTM}(x) = N(T_5 + 500) $$

for any $x \in X$. The LSTM network consists of a standard LSTM layer $L : \mathbb{R}^4 \to \mathbb{R}^4$ and a dense layer $D : \mathbb{R}^4 \to \mathbb{R}$, represented as $y = D(M(x))$. For a specified $T_{\text{th}}$ (usually $N_{\text{th}}$), if $y \geq y_{\text{th}}$, we predict the page is a bad page.

**Fig. 1: Distribution of Q—bad page.**

---

**TABLE I: Performances of bad page detectors**

<table>
<thead>
<tr>
<th>Network</th>
<th>Number of mispredicted pages</th>
<th>wasted P/E cycles (×100)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TDNN</td>
<td>375 (2.9%)</td>
<td>15982</td>
</tr>
<tr>
<td>LSTM</td>
<td>376 (2.9%)</td>
<td>16084</td>
</tr>
<tr>
<td>SVM</td>
<td>358 (2.8%)</td>
<td>19761</td>
</tr>
</tbody>
</table>

---

**REFERENCES**

