반도체디스플레이기술학회지 제21권 제3호(2022년 9월)
Journal of the Semiconductor & Display Technology, Vol. 21, No. 3. September 2022.

데이터 쓰기 패턴 분석을 통한 비휘발성 메모리 기반 딥러닝 시스템의 수명 연장 기법

최주희[*†]
[*†]상명대학교 스마트정보통신공학과

Lifetime Extension Method for Non-Volatile Memory based Deep Learning System by analyzing Data Write Pattern

Juhee Choi[*†]
[*†]Dept. of Smart Information Communication Engineering, Sangmyung University

ABSTRACT

Modern computer systems usually have special hardware for operations used in deep learning workload even edge computing environment. Non-volatile memories (NVMs) have been considered for alternative memory storage because they consume little static energy and occupy small area. However, there is a problem for NVMs to be directly adopted. An NVM cell has limited write endurance, so that the lifetime of NVM-based memory system is much shorter than that of conventional memory system. To overcome this problem for the deep learning system, this paper proposes a novel method to extend the lifetime based on the analysis of the deep learning workloads. If an incoming block has more than a predefined number of frequently used values, the cacheline is defined as write friendly block. During the victim selection, the cacheline has lower possibility to be chosen as victim. The experimental results show that the lifetime is increased by about 50% and energy consumption is decreased by 3% with a little performance hurt.

Key Words : Non-Volatile Memories, Lifetime Extension, Cache Replacement Policy

1. 서 론

최근 컴퓨터 시스템은 내부에 딥러닝을 위한 전용 하드웨어 가속기 또는 연산을 위한 별도의 명령어 셋을 가지고 있다[1,2]. 이런 하드웨어적인 지원은 서버 등과 같은 고성능 시스템에서 주로 도입되었으나, 보안이나 네트워크 환경 문제로 엣지 컴퓨팅 환경에서도 채택되고 있다[3]. 비교적 전력 소모량이나 코어(core) 면적의 중요성이 부각되지 않는 고성능 시스템에 비해 엣지 컴퓨팅 환경은 대부분 배터리에 의존하는 내장형 시스템으로 이루어져 있다. 따라서, 이런 환경에서는 성능 향상 뿐만 아니라 저전력 및 소형화도 중요하다.

[†]E-mail: jhplus@smu.ac.kr

비휘발성 메모리(non-volatile memory)는 기존의 휘발성 메모리를 대신해서 이러한 환경에 적합한 대안으로 연구되어 왔다[4,5]. 최근 공정이 고도화됨에 따라 정적 전력 소모량(static energy consumption)이 동적 전력 소모량(dynamic energy consumption)보다 커지고 있다. 이런 상황에서 정적 전력 소모량이 거의 없는 비휘발성 메모리는 코어에서 가장 많은 정적 전력을 소모하고 있는 캐시로 사용하기에 유리한 것으로 여겨지고 있다. 또한, 비휘발성 메모리는 저장 용량의 밀도가 커서 같은 면적에 더 많은 정보를 저장할 수 있으므로, 칩의 크기가 중요한 내장형 시스템에서는 비휘발성 메모리의 장점은 더욱 커진다.

그러나, 비휘발성 메모리는 쓰기 동작에서 여러가지 문제가 발생하는 단점이 있다. 데이터를 수정하기 위해서는 읽기 동작에 비해 더 많은 전력을 소모할 뿐만 아니라 동

작이 완료되기 까지의 시간 또한 길다. 또한, 쓰기 횟수의 제한이 있어 비휘발성 메모리 소자에 일정 횟수의 쓰기 동작이 일어나면 해당 셀은 마모(worn-out)이 발생해서 더 이상 정상적으로 정보를 저장할 수 없다. 따라서, 그 시점을 기준으로 전체 시스템의 실패(failure)가 발생하여, 전통적인 휘발성 메모리를 사용할 때 보다 시스템의 수명이 크게 줄어든다.

이를 극복하기 위해서, 본 논문에서는 딥러닝 워크로드를 분석해서 캐시 라인의 쓰기 동작을 줄일 수 있는 캐시 교체 정책(cache replacement policy)을 제시한다. 우선, 자주 사용되는 값들을 추출하여 별도의 공간에 저장한다. 캐시에 쓰기 접근이 발생하는 경우, 해당 캐시 라인을 쓰기 전에 자주 사용되는 값들의 개수를 파악한다. 만약 이 개수가 많은 경우, 추후에도 쓰기 동작을 재활용할 여지가 많은 캐시 라인으로 가정하여, 캐시 교체시 그 우선순위를 낮춘다. 따라서, 해당 캐시 라인은 교체 시기가 늦어지게 되고, 이를 통해 불필요한 쓰기 동작의 감소를 가져올 수 있다. 이렇게 수정된 캐시 교체 정책은 딥러닝 시스템을 위한 최신 실험 환경을 수정하여 구현한 후, 실험을 통해 성능은 거의 저하되지 않으면서도 수명은 연장되는 것을 확인할 수 있다.

2. 관련 연구

비휘발성 메모리는 정적 전력 소모량이 극히 작으면서도 저장 용량 대비 밀도가 높기 때문에, 코어에서 큰 비중을 차지하는 메인 메모리 또는 하위 레벨 캐시에서 DRAM또는 SRAM의 대체제로 연구되어 왔다. 연구자들에게 주된 관심을 받은 비휘발성 메모리 중에 하나가 Fig. 1에 표시된 Phase change memory(PCM)이다[6].

PCM은 합금형태의 소자에 정보를 저장하는 메모리이다. PCM 셀은 2가지 상태(phase)중에 하나의 상태를 가지는데, 전기저항(electrical resistivity)과 광학 반사율(optical reflectivity)을 기준으로 amorphous 상태와 crystalline 상태로 구분된다. Amorphous 상태는 RESET이라고도 하며, 일반적으로 '0'값을 표시한다. 반면, crystalline 상태는 SET이라고 불리며, '1'값을 나타낸다. PCM 셀의 상태를 바꾸기 위해서는 전류를 통해 셀에 열을 발생시킨다.

PCM은 각 셀의 상태로 비트(bit) 정보를 유지하므로, 정보 저장을 위해서 전력을 필요로 하지 않는다. 또한, 동일한 면적에 더 많은 정보 저장 용량을 구현할 수 있다. 이런 이유로 비휘발성 메모리는 면적과 전력 소모량이 중요한 내장형 시스템 연구자들에게 꾸준한 연구의 대상이 되어 왔다.

그러나, 이러한 장점들에도 불구하고, 전통적인 메모리

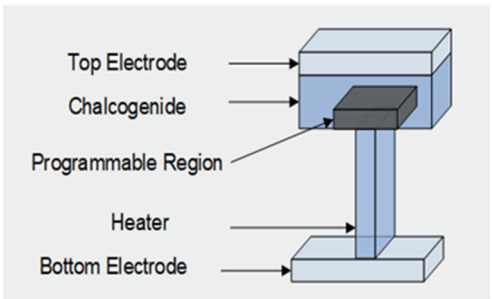

Fig. 1. Phase change memory (PCM).

를 대신해서 쉽게 사용되기 어렵다. 기존의 시스템들은 비트 정보를 읽거나 쓰는데 필요한 전력량이나 시간이 비교적 일정했으나, 비휘발성 메모리들은 읽기와 쓰기 동작에 비대칭성이 발생한다. PCM 셀의 정보를 수정하기 위해서는 많은 전력과 긴 시간을 필요로 하기 때문에, 쓰기 동작이 소모하는 전력량과 대기 시간이 늘어난다. 또한, 일정한 횟수의 상태 전이가 이루어지면, 더이상 해당 셀은 정보를 제대로 저장할 수 없는 마모의 문제가 발생한다.

이런 문제점을 극복하기 위해서, 쓰기 횟수를 줄이기 위한 여러가지 기법들이 제시되어 왔다[7-9]. 연구자들이 가장 많이 참고하는 기법 중에 하나로 Read-before-write (RBW) 기법이 존재한다[7]. 쓰기 동작이 이루어지기 전에 미리 해당 캐시 라인의 데이터를 읽어서 값이 변경될 비트만 쓰기 동작을 수행한다. 이 과정에서 불필요한 쓰기 동작을 회피함으로써, 쓰기 횟수를 크게 감소시켜 전력 소모량 감소와 수명 연장을 가져왔다.

3. 자주 사용되는 값 기반 캐시 교체 정책

3.1 캐시 쓰기 패턴 분석

기존의 캐시 관리 기법을 개선하기 위해서, 먼저 딥러닝 워크로드의 데이터 접근 패턴을 쓰기 동작 위주로 분석하였다. 실행 환경은 4장에서 구체적으로 설명한다.

캐시는 set와 way로 구분되어서 관리되는데, way의 경우는 일반적으로 데이터 접근 패턴과 무관하게 고르게 접근되므로, set의 쓰기 횟수를 관찰하였다. Fig. 2는 PCM으로 구성된 L2 캐시의 각 set별 쓰기 횟수를 정규화하고 이를 정렬한 것이다. X축은 왼쪽부터 쓰기 횟수가 많은 set부터 낮은 set까지 나열한 것이다. 이 그림에서 절대적인 set번호는 의미가 없으므로, X축의 숫자는 set의 번호가 아니라 쓰기 횟수로 정렬한 순서이다. 실제로 가장 많이 쓰기가 이루어진 set는 974과 363이지만, Fig. 2에서 가장 왼쪽 1번과 그 다음 왼쪽인 2번으로 표시되었다. 막대 그래프는 전

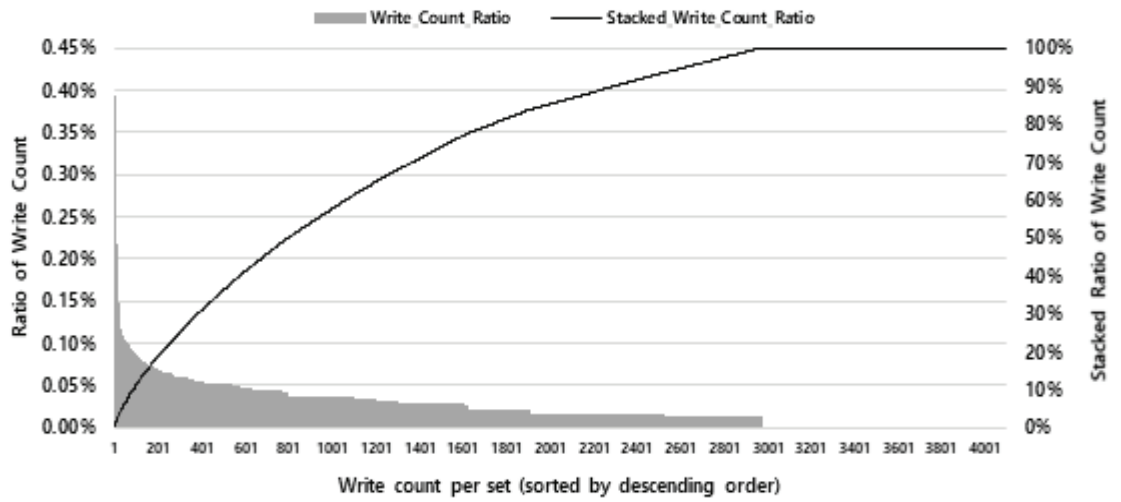

Fig. 2. Write count per set (sorted by descending order).

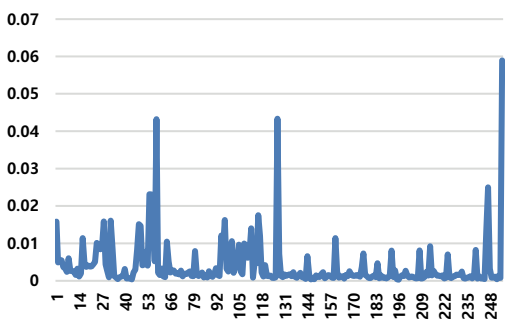

Fig. 3. Normalized write counts per value from 1 to 255.

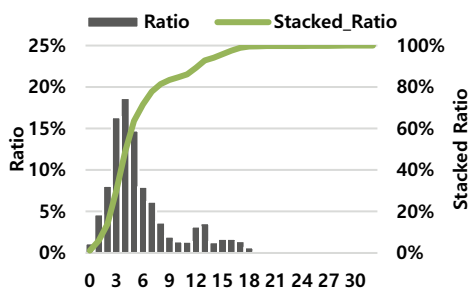

Fig. 4. Ratio and Stacked ratio of frequently used values across a cacheline.

체 쓰기 횟수를 기준으로 정규화한 값을 나타낸 것으로 왼쪽 Y축의 척도를 사용한다. 실선은 누적된 쓰기 횟수로 오른쪽 Y축 척도에 대응된다. 쓰기 횟수가 많은 10개까지의 내용은 확대하여 Fig.2내부에 별도로 표시하였다.

먼저, Fig. 2에서 확인할 수 있는 것은 각 set별로 쓰기 횟수의 차이가 크다는 것이다. 전체 set의 개수는 4096개이지만, 가장 많이 쓰기가 이루어진 set과 가장 적은 쓰기가 이루어진 set은 100배이상의 차이를 보인다. 누적 쓰기 횟수를 보면, 800번째 set까지의 쓰기 횟수가 이미 전체 쓰기 횟수의 50%를 차지하는 것을 확인할 수 있다. 그리고, 확대된 그림에서 볼 수 있듯이 가장 쓰기 횟수가 많은 2개의 set은 3번째 쓰기 횟수가 많은 set에 비해 2배이상 쓰기 접근이 이루어지고 있다.

더 세부적인 데이터 쓰기 패턴 분석을 위해서 바이트(byte) 단위로는 어떤 값들이 많이 쓰여지는지 그 빈도를 분석하였다. Fig. 3은 전체 쓰기 횟수를 기준으로 각 값들의 쓰기 횟수를 정규화하여 표시한 것이다. 1 바이트로 나타낼 수 있는 값이 2^8개가 있으므로 X축에는 1에서 255까지 순서대로 나열하였다. 이 그림에서 0값은 빠져있는데, 0값의 경우는 다른 값들에 비해서 압도적으로 많이 쓰여지고 있으나, 이미 기존의 연구들에서 0값의 쓰기 횟수를 줄이기 위한 기법들이 많이 연구되었으므로[10,11], 본 논문에서 이미 이런 기법들이 사용되고 있다고 가정하고, 분석에서 제외하였다. Fig. 3에서 볼 수 있듯이, 0값을 제외하더라도 특정한 몇몇 값들이 다른 값들에 비해서 많이 쓰여지는 것을 볼 수 있다.

좀 더 입체적인 분석을 위해서, 하나의 캐시 라인에서 Fig. 3에서 나타난 자주 쓰이는 값들 중 상위 10개의 값이 몇개나 나타나는지 개수를 Fig. 4에 표시하였다. X축은 상위 10개값 중 하나의 캐시 라인에 몇개가 나타나는지를

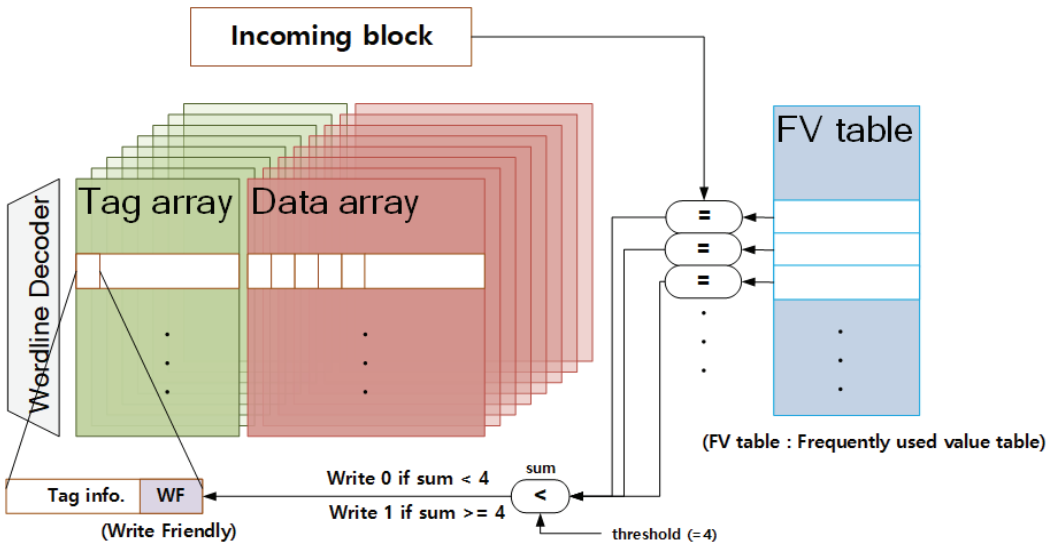

Fig. 5. Overall structure of frequently value based cache replacement policy (FV).

표시하는 것이다. 좌측 X축에서 18%라는 의미는 전체 캐시 라인 중에서 18%의 캐시 라인이 상위 10개 값들 중 4개의 값을 가지고 있다는 뜻이다. 그리고, 우측 Y축은 누적된 비율을 의미한다. 바꾸어 말하면, X축의 4의 값에 우측 Y축이 60%인 것은 전체 캐시 라인 중 상위 10개의 값과 일치하는 바이트의 수가 4이하인 캐시 라인이 60%라는 뜻이다. 따라서, 40%이상의 캐시 라인들은 자주 쓰이는 값들을 4바이트 이상 포함하고 있다고 볼 수 있다.

3.2 자주 사용되는 값 기반 캐시 교체 정책

위의 분석 결과를 바탕으로 본 논문에서는 전력 소모량 감소 및 수명 연장을 위한 새로운 캐시 관리 기법인 자주 사용되는 값 기반(Frequently used value, FV) 기반 캐시 교체 정책을 제시한다. 캐시에 쓰기 동작이 일어나기 전에 미리 설정된 값들과 캐시 라인의 값들을 바이트 단위로 비교하여 일치하는 개수를 파악한다. 일치하는 개수가 일정한 수를 넘는 경우, 자주 쓰이는 캐시 라인으로 지정해서, 추후 캐시 미스(cache miss)가 발생하여 희생(victim) 캐시 라인을 찾아낼 때, 해당 캐시 라인의 우선 순위를 낮추어서, 최대한 오랫 동안 캐시에 유지되도록 한다.

Fig. 5는 FV를 위해 도입된 데이터 구조를 보여주고 있다. FV 테이블에는 앞에서 분석한 자주 사용되는 값들을 저장하고 있다. 새로운 캐시 라인(incoming block)이 갱신되는 경우, FV 테이블의 값들과 비교해서 몇개가 일치하는지를 계산한다. 그리고, 해당 값이 지정된 값보다 큰 경우 해당 캐시 라인은 쓰기 친화적인(write friendly, WF) 캐시 라인으로 지정한다. 쓰기 친화도를 표시하는 비트(WF bit)는

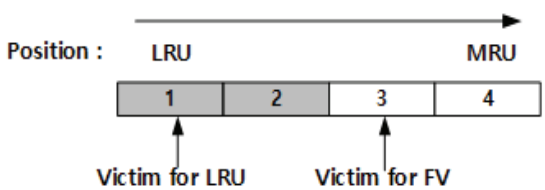

Fig. 6. Example of FV replacement policy.

기존의 태그 구조(tag array)에 추가된다. 본 논문에서는 4보다 크거나 같은 경우를 쓰기 친화적인 캐시 라인이라고 분류한다.

캐시 미스가 발생해서 희생 캐시 라인을 찾아야 하는 경우, 우선 기존의 LRU기반의 알고리즘에 따라 캐시 라인을 탐색한다. 해당 캐시 라인이 쓰기 친화적이지 않은 경우, 희생 캐시 라인으로 선택한다. 그러나, WF비트의 값이 1인 경우는 해당 캐시 라인은 선택하지 않는다. Fig. 6에 예시를 보여주고 있다. 그림에서 회색 음영이 쓰기 친화적인 캐시 라인이다. 원래 캐시 미스가 발생했을 때는 1번 캐시 라인이 희생 캐시 라인으로 선택되어야 하지만, WF비트가 1로 설정된 1번과 2번 캐시 라인은 건너뛰고, 3번 캐시 라인이 선택된다.

FV를 도입하는 경우, 두 가지 고려해야 할 문제가 있다. 먼저, FV 테이블의 크기와 WF비트로 인해 증가된 면적이다. FV 테이블 크기는 1바이트 값 10개를 저장하므로 10B이고, WF비트의 경우 기존의 태그 비트에 추가되므로, 4KB가 추가된다. L2 캐시의 크기가 2MB인 점을 고려하면, 무시할 정도의 면적만이 추가된다. 그러나, 면적 문제와 달리

캐시 실패율(cache miss rate)의 증가는 무시할 수 없다. 기존의 LRU는 자주 사용되는 캐시 라인을 최대한 보존하는 정책인 반면에, FV의 경우, 멀지 않은 시점에서 사용될 캐시 라인이 교체될 수 있으므로, 캐시 미스가 증가하고, 이것이 성능상의 저하를 유발할 수 있다. 따라서, 실험을 통해 성능을 확인할 필요가 있다.

4. 실험 환경 및 결과

4.1 실험 환경

이 장에서는 본 논문에서 제안한 캐시 구조의 효율성을 검증하기 위해서 시뮬레이션을 진행하였다. cycle 단위의 정확성을 가지면서도 사용자 환경에서 성능 평가가 가능한 end-to-end full stack 시뮬레이터인 SMAUG를 사용하였다[12]. 실험에 사용된 시스템의 L1 명령어 캐시와 L1 데이터 캐시가 각각 64KB이며, 공용 L2 캐시는 2MB이다. L1 캐시들은 SRAM이고, L2 캐시는 PCM이고, 구체적인 시스템 실험 환경은 Table 1에 표시하였다. 실험을 위한 워크로드는Minerva[13]를 선택하였다.

본 논문의 전력 소모 감소량과 수명 연장을 비교하기 위해서 그동안 비휘발성 메모리 연구자들에게 가장 많은 인용되었던 기법 중에 하나인 RBW와 DI16를 같이 실험하고, 비교하여 결과를 표시하였다[7].

4.2 실험 결과

Fig. 7은 RBW와 DI16기법에 LRU와 본 논문에서 제안한 FV를 결합하여 시뮬레이션을 수행한 후, 정규화된 전력 소모량과 수명을 나타낸 것이다. X축은 실험한 기법의 종류를 표시한다. FV(RBW)는 RBW기법에 FV를 도입한 것이다. 좌측 Y축은 막대 그래프의 척도이며, 정규화된 전력 소모량으로 LRU(RBW)를 기준으로 하였다. 우측 Y축은 실선의 척도이며, 상대적인 수명을 나타낸 것으로 역시 LRU(RBW)의 수명을 기준으로 삼았다. 기존의 연구 결과와 유사하게 RBW보다 DI16이 소량이나마 전력 소모량이 많은 반면, 수명도 조금 연장되었다. RBW에서 LRU와 FV의 전력 소모량의 차이는 5%이지만, 수명은 41%정도 늘어났다. DI16의 경우, LRU에 비해 FV가 전력 소모량도 3%정도 감소하고, 수명은 50%이상 증가하였다.

Table 1. System configurations

Parameter	Value
Core Type	X86, out-of-order, 2.5GHz
Function Units	6 IALU, 2 IMULT, 4 FPALU, 2 FPMULT, branch predictors
I-Cache	64KB, 4-way, 32B, 2 cycles
D-Cache	64KB, 4-way, 32B, 2 cycles
L2 Cache	2MB, 16-way, 64B, 30 cycles

Fig. 8은 새로운 캐시 교체 정책에 따른 손실을 확인하기 위해서, 정규화된 MPKI(Miss per Kilo Instructions)와 실행 시간을 구한 것이다. 캐시 적중률(cache hit ratio)에 최적화된 LRU를 채택한 캐시에 비해, FV 기법을 활용한 캐시의 캐시 미스가 늘어나는 것을 확인할 수 있다. 다만, FV 기법을 채택한 캐시에서 실패가 늘어나는 비율은 RBW와 DI16이 각각 3%와 5%인 반면, 실제 성능의 하락 비율은 각각0.7%와 0.5%로 상대적으로 줄어든다. 이것은 L2 캐시 실패의 경우, prefetch나 MSHRs(Miss status holding registers)과 같이 지연 시간을 줄이는 다양한 기법들이 존재하므로, 캐시 라인을 다시 가져오는 시간의 일부가 숨겨지기 때문이다.

따라서, DI16기법에 FV을 도입한 결과, 성능의 저하는 거의 없으면서, 전력 소모량도 3% 줄이면서 시스템 전체의 수명은 50%이상 늘어났음을 확인할 수 있다.

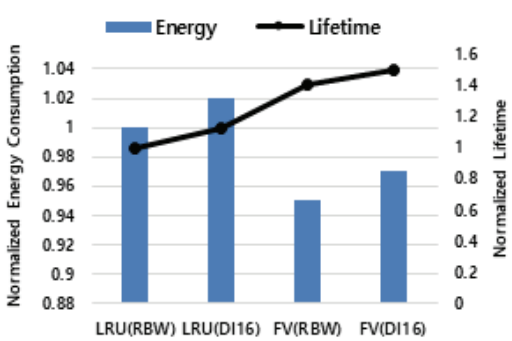

Fig. 7. Normalized energy consumption and lifetime.

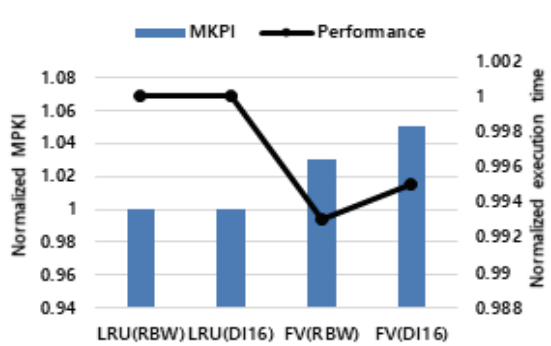

Fig. 8. Normalized MPKI and execution time(performance).

5. 결 론

본 논문에서는 딥러닝 시스템에서 비휘발성 메모리를 사용한 캐시의 전력 소모량 및 수명 연장을 위한 FV기법을 제안하였다. 딥러닝 워크로드를 분석하여, 일부 set에서 쓰기가 집중되는 것을 확인하였으며, 캐시 라인의 바이트별 값들이 통계적으로 불균형하게 사용되고 있음을 보였다.

그리고, 이를 활용하기 위해서, 캐시 라인에 쓰기가 발생할 때, 자주 사용되는 값들의 개수를 세고, 이를 바탕으로 쓰기 횟수를 줄일 수 있는 캐시 라인인지 아닌지 여부를 기록한다. 추후 희생 캐시 라인을 선택할 때, 교체될 때 쓰기 횟수를 줄일 여지가 있는 수 캐시 라인의 우선순위를 낮추어서, 최대한 해당 캐시 라인이 캐시에 존재하도록 유도하였다.

비휘발성 메모리 연구에서 많이 사용되는 기존의 기법에 LRU와 FV를 구현하여 실험한 결과, 수행 시간의 증가는 거의 없으면서, 3%의 전력 소모량 감소와 1.5배이상의 수명 연장을 확인할 수 있었다.

감사의 글

본 연구는 2021년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받은 기초연구사업 연구임(NRF-2021R1G1A1004340).

참고문헌

1. Jinwook Song et al., “7.1 An 11.5TOPS/W 1024-MAC Butterfly Structure Dual-Core Sparsity-Aware Neural Processing Unit in 8nm Flagship Mobile SoC,” 2019 IEEE International Solid-State Circuits Conference - (ISSCC), pp. 130-132, 2019.
2. Shaoli Liu et al., “Cambricon: An Instruction Set Architecture for Neural Networks,” 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), pp. 393-405, 2016.
3. Jiasi Chen and Xukan Ran, “Deep Learning With Edge Computing: A Review,” in Proceedings of the IEEE, Vol. 107, No. 8, pp. 1655-1674, Aug. 2019.
4. Ju Young Jeong, “Feasibility Study of Non-volatile Memory Device Structure for Nanometer MOSFET,” Journal of the Semiconductor & Display Technology, Vol. 14, No. 2, pp. 41-45, 2015.
5. Tae Hyun Kim, Yang On, and Sang Yeon Jun, “Design of Asynchronous Non-Volatile Memory Module Using NAND Flash Memory and PSRAM,” Journal of the Semiconductor & Display Technology Vol. 19, No.3, pp. 118-123, 2020.
6. H.-S. Philip Wong et al., “Phase Change Memory,” in Proceedings of the IEEE, Vol. 98, No. 12, pp. 2201-2227, Dec. 2010.
7. Yongsoo Joo, Dimin Niu, Xiangyu Dong, Guangyu Sun, Naehyuck Chang, and Yuan Xie, “Energy- and endurance-aware design of phase change memory caches,” 2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010), pp. 136-141, 2010.
8. Alexandre P. Ferreira, Miao Zhou, Santiago Bock, Bruce Childers, Rami Melhem, and Daniel Mossé, “Increasing PCM main memory lifetime,” 2010 Design, Automation & Test in Europe Conference & Exhibition (DATE 2010), pp. 914-919, 2010.
9. Shihao Song, Anup Das, Onur Mutlu, and Nagarajan Kandasamy, “Improving phase change memory performance with data content aware access,” In: Proceedings of the 2020 ACM SIGPLAN International Symposium on Memory Management. pp. 30-47, 2020.
10. Guangshan Duan and Shuai WangG, “Exploiting narrow-width values for improving non-volatile cache lifetime,” 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1-4, 2014.
11. Nezam Rohbani et al, “NVDL-Cache: Narrow-Width Value Aware Variable Delay Low-Power Data Cache,” 2019 IEEE 37th International Conference on Computer Design (ICCD), pp. 264-272, 2019.
12. Sam Xi et al, “SMAUG: End-to-end full-stack simulation infrastructure for deep learning workloads,” ACM Transactions on Architecture and Code Optimization (TACO), Vol. 17, No. 4, pp. 1-26, 2020.
13. Brandon Reagen et al., “Minerva: Enabling Low-Power, Highly-Accurate Deep Neural Network Accelerators,” 2016 ACM/IEEE 43rd Annual International Symposium on Computer Architecture (ISCA), pp. 267-278, 2016.

접수일: 2022년 7월 25일, 심사일: 2022년 9월 6일,
게재확정일: 2022년 9월 19일