



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0166028
(43) 공개일자 2022년12월16일

(51) 국제특허분류(Int. Cl.)
G06N 5/02 (2006.01) G06N 3/063 (2006.01)
G06N 3/08 (2006.01)
(52) CPC특허분류
G06N 5/02 (2019.01)
G06F 13/1615 (2013.01)
(21) 출원번호 10-2021-0074809
(22) 출원일자 2021년06월09일
심사청구일자 없음

(71) 출원인
삼성전자주식회사
경기도 수원시 영통구 삼성로 129 (매탄동)
울산과학기술원
울산광역시 울주군 언양읍 유니스트길 50
(72) 발명자
전명재
울산광역시 울주군 언양읍 유니스트길 50
박찬호
울산광역시 울주군 언양읍 유니스트길 50
이규호
울산광역시 울주군 언양읍 유니스트길 50
(74) 대리인
특허법인 무한

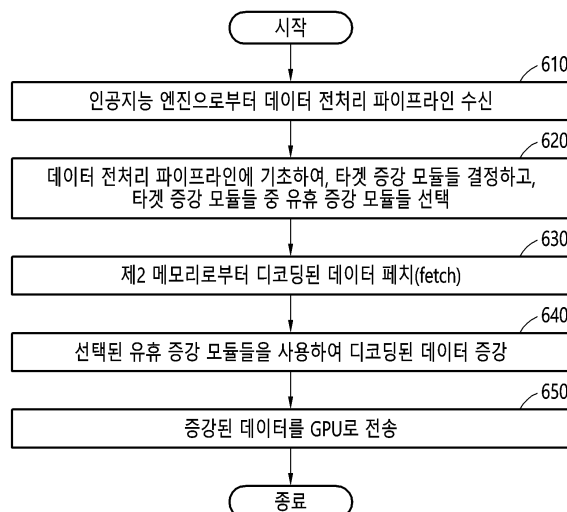
전체 청구항 수 : 총 13 항

(54) 발명의 명칭 데이터 전처리를 위한 스토리지 장치 및 그 동작 방법

(57) 요약

스토리지 장치에서 수행되는 데이터 전처리를 수행하는 방법 및 장치가 개시된다. 일 실시예에 따른 데이터 전처리(pre-processing)를 위한 스토리지 장치는 전처리할 로우(raw) 데이터를 저장하는 제1 메모리; 재구성가능한(re-configurable) 하나 이상의 증강 모듈(augmentation module)들이 프로그래밍되고, 로우 데이터를 디코딩하는 디코더(decoder), 디코딩된 데이터를 저장하는 제2 메모리 및 프로세서를 포함하는 FPGA(Field Programmable Gate Array)를 포함하고, 프로세서는 데이터 전처리 파이프라인(pipeline)에 기초하여, 하나 이상의 증강 모듈(augmentation module)들 중 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정하고, 타겟 증강 모듈들 중 유헴(idle) 증강 모듈들을 사용하여 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하고, 증강된(augmented) 데이터를 GPU(graphic processing unit)로 전송할 수 있다.

대 표 도 - 도6



(52) CPC특허분류

G06N 3/063 (2013.01)

G06N 3/08 (2013.01)

명세서

청구범위

청구항 1

데이터 전처리(pre-processing)를 위한 스토리지 장치에 있어서,

전처리할 로우(raw) 데이터를 저장하는 제1 메모리;

재구성가능한(re-configurable) 하나 이상의 증강 모듈(augmentation module)들이 프로그래밍되고, 상기 로우 데이터를 디코딩하는 디코더(decoder), 디코딩된 데이터를 저장하는 제2 메모리 및 프로세서를 포함하는 FPGA(Field Programmable Gate Array)

를 포함하고,

상기 프로세서는,

데이터 전처리 파이프라인(pipeline)에 기초하여, 상기 하나 이상의 증강 모듈(augmentation module)들 중 상기 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정하고, 상기 타겟 증강 모듈들 중 유ힴ(idle) 증강 모듈들을 사용하여 상기 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하고, 상기 증강된(augmented) 데이터를 GPU(graphic processing unit)로 전송하는,

스토리지 장치.

청구항 2

제1항에 있어서,

상기 프로세서는,

상기 제2 메모리에 상기 디코딩된 데이터가 존재하지 않는 경우, 상기 제1 메모리에 저장된 상기 로우 데이터를 디코딩하여 상기 제2 메모리에 더 저장하는,

스토리지 장치.

청구항 3

제1항에 있어서,

상기 프로세서는,

상기 증강된 상기 데이터를 호스트(host)를 우회(bypass)하여 상기 GPU로 전송하는,

스토리지 장치.

청구항 4

제1항에 있어서,

상기 프로세서는,

상기 데이터 전처리 파이프라인을 상기 타겟 증강 모듈들에 기초하여 파이프라인 병렬화(pipeline parallelization)하고,

상기 타겟 증강 모듈들의 유ힴ(idle) 상태에 대한 정보를 수신하고,

상기 타겟 증강 모듈들 중 유틸 증강 모듈들을 선택하는,
스토리지 장치.

청구항 5

제1항에 있어서,
상기 프로세서는,
상기 디코딩된 데이터를 상기 유틸 증강 모듈들로 병렬 처리하는,
스토리지 장치.

청구항 6

제1항에 있어서,
상기 선택된 유틸 증강 모듈들을 사용하여 상기 디코딩된 데이터를 증강(augment)할 때마다 생성되는 중간 데이터를 저장하는 제3 메모리
를 더 포함하는,
스토리지 장치.

청구항 7

스토리지 장치에서 수행되는 데이터 전처리(pre-processing) 방법에 있어서,
데이터 전처리 파이프라인(pipeline)에 기초하여, 하나 이상의 증강 모듈(augmentation module)들 중 상기 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정하고, 상기 타겟 증강 모듈들 중 유틸(idle) 증강 모듈들을 선택하는 동작;
상기 선택된 유틸 증강 모듈들을 사용하여 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하는 동작; 및
상기 증강된(augmented) 데이터를 GPU(graphic processing unit)로 전송하는 동작
을 포함하는,
데이터 전처리 방법.

청구항 8

제7항에 있어서,
상기 제2 메모리에 상기 디코딩된 데이터가 존재하지 않는 경우, 제1 메모리에 저장된 로우 데이터를 디코딩하여 상기 제2 메모리에 저장하는 동작
을 더 포함하는,
데이터 전처리 방법.

청구항 9

제7항에 있어서,
상기 증강된 데이터를 GPU(graphic processing unit)로 전송하는 동작은,

호스트(host)를 우회(bypass)하여 상기 GPU로 전송하는 동작
을 포함하는,
데이터 전처리 방법.

청구항 10

제7항에 있어서,
데이터 전처리 파이프라인(pipeline)에 기초하여, 하나 이상의 증강 모듈(augmentation module)들 중 상기 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정하고, 상기 타겟 증강 모듈들 중 유휴(idle) 증강 모듈들을 선택하는 동작은,
상기 데이터 전처리 파이프라인을 상기 타겟 증강 모듈들에 기초하여 파이프라인 병렬화(pipeline parallelization)하는 동작;
상기 타겟 증강 모듈들의 유휴(idle) 상태에 대한 정보를 수신하는 동작; 및
상기 타겟 증강 모듈들 중 유휴 증강 모듈들을 선택하는 동작
을 포함하는,
데이터 전처리 방법.

청구항 11

제7항에 있어서,
상기 선택된 유휴 증강 모듈들을 사용하여 상기 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하는 동작은,
상기 디코딩된 데이터를 상기 유휴 증강 모듈들로 병렬 처리하는 동작
을 포함하는,
데이터 전처리 방법.

청구항 12

제7항에 있어서,
상기 선택된 유휴 증강 모듈들을 사용하여 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하는 동작은,
상기 선택된 유휴 증강 모듈들을 사용하여 상기 디코딩된 데이터를 증강(augment)할 때마다 생성된 중간 데이터를 제3 메모리에 저장하는 동작
을 더 포함하는,
데이터 전처리 방법.

청구항 13

하드웨어와 결합되어 제7항의 방법을 실행시키기 위하여 컴퓨터 판독 가능한 기록매체에 저장된 컴퓨터 프로그램.

발명의 설명

기술 분야

[0001] 아래 실시예들은 데이터 전처리를 위한 스토리지 장치 및 그 동작 방법에 관한 것이다.

배경 기술

[0003] 인공지능 학습의 데이터 전처리(data pre-processing) 과정이 전체 학습 시간에서 차지하는 비율이 높아짐에 따라, 전처리 과정 가속화의 중요성이 높아지고 있다. 2020년도 MLSys에서 발표된 “Systematic Methodology for Analysis of Deep Learning Hardware and Software Platforms” 연구에 따르면, 데이터 전처리 파이프라인이 전체 학습 시간의 50~60%까지 차지할 수 있다.

[0004] 또한, 데이터 전처리 파이프라인은 전체 학습 결과의 정확도에 영향을 줄 수 있다. 2017년도 CVPR에 발표된 “Densely Connected Convolutional Networks” 연구에 따르면 데이터 증강 기법으로 이동(Transition)과 대칭(Mirroring)을 사용한 경우, CIFAR-100을 학습한 DenseNet 모델의 정확도가 최소 3%에서 최대 11% 이상 증가했다는 분석 결과가 있다. 아울러, 인공지능 모델의 정확도를 높일 수 있는 복잡하고 견고한 데이터 증강 방법이 지속해서 개발되고 있다.

[0005] 전술한 배경기술은 발명자가 본원의 개시 내용을 도출하는 과정에서 보유하거나 습득한 것으로서, 반드시 본 출원 전에 일반 공중에 공개된 공지기술이라고 할 수는 없다.

발명의 내용

해결하려는 과제

과제의 해결 수단

[0007] 일 실시예에 따른 데이터 전처리(pre-processing)를 위한 스토리지 장치는, 전처리할 로우(raw) 데이터를 저장하는 제1 메모리; 재구성가능한(re-configurable) 하나 이상의 증강 모듈(augmentation module)들이 프로그래밍되고, 상기 로우 데이터를 디코딩하는 디코더(decoder), 디코딩된 데이터를 저장하는 제2 메모리 및 프로세서를 포함하는 FPGA(Field Programmable Gate Array)를 포함하고, 상기 프로세서는, 데이터 전처리 파이프라인(pipeline)에 기초하여, 상기 하나 이상의 증강 모듈(augmentation module)들 중 상기 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정하고, 상기 타겟 증강 모듈들 중 유휴(idle) 증강 모듈들을 사용하여 상기 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하고, 상기 증강된(augmented) 데이터를 GPU(graphic processing unit)로 전송할 수 있다.

[0008] 일 실시 예에 따르면 상기 프로세서는, 상기 제2 메모리에 상기 디코딩된 데이터가 존재하지 않는 경우, 상기 제1 메모리에 저장된 상기 로우 데이터를 디코딩하여 상기 제2 메모리에 더 저장할 수 있다.

[0009] 일 실시 예에 따르면 상기 프로세서는, 상기 증강된 상기 데이터를 호스트(host)를 우회(bypass)하여 상기 GPU로 전송할 수 있다.

[0010] 일 실시 예에 따르면 상기 프로세서는, 상기 데이터 전처리 파이프라인을 상기 타겟 증강 모듈들에 기초하여 파이프라인 병렬화(pipeline parallelization)하고, 상기 타겟 증강 모듈들의 유휴(idle) 상태에 대한 정보를 수신하고, 상기 타겟 증강 모듈들 중 유휴 증강 모듈들을 선택할 수 있다.

[0011] 일 실시 예에 따르면 상기 프로세서는, 상기 디코딩된 데이터를 상기 유휴 증강 모듈들로 병렬 처리할 수 있다.

[0012] 일 실시 예에 따르면 상기 스토리지 장치는, 상기 선택된 유휴 증강 모듈들을 사용하여 상기 디코딩된 데이터를 증강(augment)할 때마다 생성되는 중간 데이터를 저장하는 제3 메모리를 더 포함할 수 있다.

[0013] 일 실시 예에 따른 스토리지 장치에서 수행되는 데이터 전처리(pre-processing) 방법은, 데이터 전처리 파이프

라인(pipeline)에 기초하여, 하나 이상의 증강 모듈(augmentation module)들 중 상기 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정하고, 상기 타겟 증강 모듈들 중 유휴(idle) 증강 모듈들을 선택하는 동작; 상기 선택된 유휴 증강 모듈들을 사용하여 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하는 동작; 및 상기 증강된(augmented) 데이터를 GPU(graphic processing unit)로 전송하는 동작을 포함할 수 있다.

[0014] 일 실시 예에 따른 데이터 전처리 방법은, 상기 제2 메모리에 상기 디코딩된 데이터가 존재하지 않는 경우, 제1 메모리에 저장된 로우 데이터를 디코딩하여 상기 제2 메모리에 저장하는 동작을 더 포함할 수 있다.

[0015] 일 실시 예에 따른 상기 증강된 데이터를 GPU(graphic processing unit)로 전송하는 동작은, 호스트(host)를 우회(bypass)하여 상기 GPU로 전송하는 동작을 포함할 수 있다.

[0016] 일 실시 예에 따른 데이터 전처리 파이프라인(pipeline)에 기초하여, 하나 이상의 증강 모듈(augmentation module)들 중 상기 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정하고, 상기 타겟 증강 모듈들 중 유휴(idle) 증강 모듈들을 선택하는 동작은, 상기 데이터 전처리 파이프라인을 상기 타겟 증강 모듈들에 기초하여 파이프라인 병렬화(pipeline parallelization)하는 동작; 상기 타겟 증강 모듈들의 유휴(idle) 상태에 대한 정보를 수신하는 동작; 및 상기 타겟 증강 모듈들 중 유휴 증강 모듈들을 선택하는 동작을 포함할 수 있다.

[0017] 일 실시 예에 따른 상기 선택된 유휴 증강 모듈들을 사용하여 상기 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하는 동작은, 상기 디코딩된 데이터를 상기 유휴 증강 모듈들로 병렬 처리하는 동작을 포함할 수 있다.

[0018] 일 실시 예에 따른 상기 선택된 유휴 증강 모듈들을 사용하여 제2 메모리에 저장된 디코딩된 데이터를 증강(augment)하는 동작은, 상기 선택된 유휴 증강 모듈들을 사용하여 상기 디코딩된 데이터를 증강(augment)할 때 마다 생성된 중간 데이터를 제3 메모리에 저장하는 동작을 더 포함할 수 있다.

도면의 간단한 설명

[0020] 도 1a는 일 실시예에 따른 인공지능 학습을 위한 데이터 전처리 과정의 흐름을 도시한 도면이다.

도 1b는 인공지능 학습을 위한 데이터 전처리 과정의 증강(augmentation)을 설명하기 위한 도면이다.

도 2는 일 실시 예에 따른 데이터 전처리를 위한 스토리지 장치(101)의 구성을 나타낸 블록도이다.

도 3은 일 실시 예에 따른 하나 이상의 증강 모듈들 중 타겟 증강 모듈들이 결정되는 동작을 설명하기 위한 도면이다.

도 4는 단일 파이프라인에 따른 데이터 전처리 시 병렬 처리로 인한 동작을 설명하기 위한 도면이다.

도 5는 여러 파이프라인에 따른 데이터 전처리 시 병렬 처리로 인한 동작을 설명하기 위한 도면이다.

도 6은 일 실시 예에 따른 스토리지 장치의 동작 방법을 설명하기 위한 흐름도이다.

도 7은 일 실시 예에 따른 데이터 전처리 파이프라인에 기초하여 유휴 증강 모듈들을 선택하는 동작을 설명하기 위한 흐름도이다.

도 8은 일 실시 예에 따른 디코딩된 데이터를 페치(fetch)하는 동작을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

[0021] 실시예들에 대한 특정한 구조적 또는 기능적 설명들은 단지 예시를 위한 목적으로 개시된 것으로서, 다양한 형태로 변경되어 구현될 수 있다. 따라서, 실제 구현되는 형태는 개시된 특정 실시예로만 한정되는 것이 아니며, 본 명세서의 범위는 실시예들로 설명한 기술적 사상에 포함되는 변경, 균등물, 또는 대체물을 포함한다.

[0022] 제1 또는 제2 등의 용어를 다양한 구성요소들을 설명하는데 사용될 수 있지만, 이런 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 해석되어야 한다. 예를 들어, 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소는 제1 구성요소로도 명명될 수 있다.

[0023] 어떤 구성요소가 다른 구성요소에 "연결되어" 있다고 언급된 때에는, 그 다른 구성요소에 직접적으로 연결되어

있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 할 것이다.

[0024] 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 명세서에서, "포함하다" 또는 "가지다" 등의 용어는 설명된 특징, 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것이 존재함으로써 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성요소, 부분품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.

[0025] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 해당 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가진다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 의미를 갖는 것으로 해석되어야 하며, 본 명세서에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.

[0026] 이하, 실시예들을 첨부된 도면들을 참조하여 상세하게 설명한다. 첨부 도면을 참조하여 설명함에 있어, 도면 부호에 관계없이 동일한 구성 요소는 동일한 참조 부호를 부여하고, 이에 대한 중복되는 설명은 생략하기로 한다.

[0028] <인공지능 학습에서의 데이터 전처리 과정>

[0029] 도 1a는 일 실시예에 따른 인공지능 학습을 위한 데이터 전처리 과정의 흐름을 도시한 도면이다.

[0030] 도 1a를 참조하면, DNN(deep neural network) 학습을 위한 이미지 데이터의 전처리 과정이 도시되어 있다. 이미지는 로더(loader)로부터, 예를 들어 JPEG 형식의 파일로 페치(fetch)(103)되어 CPU(central processing unit)에 의해 전처리(104)될 수 있고, 이미지에 대한 라벨(label)은 인공지능 학습을 위한 GPU(graphic processing unit)로 전송될 수 있다.

[0031] 일 실시 예에 따르면, 데이터 전처리 과정(104)에는 디코딩(105), 사이즈 재설정(resize)(107) 및 증강(augment)(109)이 포함될 수 있다. 증강(109) 과정은 인공지능 학습의 정확도 향상을 위해 데이터를 늘리는 과정으로, 전처리 과정에서 차지하는 비중이 커짐에 따라 더욱 중요해지고 있다. 다양한 증강 기법에 대한 예시적인 설명은 도 1b를 참조하여 설명된다.

[0032] 데이터 전처리 과정은 도 1a에 도시된 바와 같이 CPU를 기반으로 수행될 수 있다. 다만 인공지능 서버에서 CPU 대비 GPU 하드웨어 성능이 증가하고, 인공지능 모델의 정확도 향상을 위한 데이터 전처리 과정의 복잡도가 증가함에 따라 고성능 모델 학습에서 기존 CPU 기반 데이터 전처리 수행은 병목단계(bottleneck)가 될 수 있다.

[0033] 단일 서버 내의 CPU와 GPU의 하드웨어 성능은 차이는 약 10배 가까이 차이가 날 수 있으며, CPU의 성능은 매년 1.1배씩 향상되는데 반대 GPU의 성능은 매년 1.5배씩 향상되고 있다. 특히, CPU에서 전처리된 데이터가 GPU에 의해 학습되기 때문에 데이터 전처리 과정 파이프라인의 지연이 발생하면, 고성능 인공지능 서버환경에서는 GPU 성능에 무관하게 학습 시간이 증가할 수 있다. 연구에 따르면, 동일 모델 기준 256개의 TPU(tensor processing unit) 서버에서 데이터 전처리 과정이 전체 학습 시간의 98%까지 차지할 수 있다는 예측이 있다. 이에 따라 CPU 기반의 데이터 전처리가 아닌, GPU(graphic processing unit) 또는 FPGA(field programmable gate array) 기반의 데이터 전처리 방법이 제시되고 있다.

[0034] GPU 기반 데이터 전처리 방법은 파이프라인의 얼마만큼을 GPU로 옮길지 결정하여 데이터 전처리 과정의 일부를 유틸 GPU 리소스를 사용하여 수행할 수 있다. 다만 이는 GPU에 유틸 리소스가 존재하는 경우에만 효율적이고, GPU 사용량이 80% 이상인 경우에는 오히려 학습 성능이 떨어질 수 있다.

[0035] FPGA(field programmable gate array) 기반 데이터 전처리 방법은 FPGA 내에 정적으로 구현된 데이터 전처리 모듈들과 파이프라인(pipeline)을 통해 데이터를 전처리할 수 있다. 다만 최근 다양한 새로운 데이터 증강 알고리즘(algorithm)이 개발됨에 따라, 데이터 증강에 사용되는 방법이 급증하고 있다. 예를 들어, 보편적으로 사용되는 인공지능 플랫폼에서 지원되는 데이터 증강 기법은 14가지 이상이고, 각 학습 작업은 최소 2개 이상의 데이터 증강 기법을 조합하여 사용하므로 실질적으로 수많은 데이터 증강 파이프라인이 존재할 수 있다. 이처럼 다양한 조합의 데이터 증강 방법에 대해 정적으로 FPGA 시스템을 미리 설계하는 것은 어려울 수 있다.

[0036] 일 실시 예에 따르면, 데이터 전처리 파이프라인이 동적으로 재구성(reconfiguration)될 수 있는 FPGA 기반 스토리지 장치가 개시된다. 일 실시 예에 따른 FPGA 기반 스토리지 장치는 데이터 전처리 과정(104)을 수행할 수

있고, 구체적으로 증강 과정(109)을 동적으로 구성할 수 있다. 일 실시 예에 따른 FPGA 기반 스토리지 장치는 도 2에서 상세히 설명된다.

- [0038] 도 1b는 인공지능 학습을 위한 데이터 전처리 과정의 증강(augmentation)을 설명하기 위한 도면이다.
- [0039] 도 1b를 참조하면, 오리지날(original) 이미지 및 증강된(augmented) 이미지들이 도시되어 있다. 다양한 실시 예들에 따르면, 도 1b에 도시된 증강 기법들로는, 일부를 잘라내는 Crop, 대칭시키는 Symmetry, 회전시키는 Rotation, 크기를 바꾸는 Scale, 노이즈가 있는 Noise, 색조를 변경하는 Hue, 일부를 가린 Obstruction, 블러 처리하는 Blur가 있다.
- [0040] 인공지능 학습을 위해선 다양한 입력 데이터가 필요하고, 특히 하나의 오리지날 데이터에 대해 여러 증강 기법이 적용된 데이터가 입력되면 인공지능 학습 결과의 정확도가 향상될 수 있다. 도 1b에 도시된 증강 기법들은 예시적인 것이고, 다양한 증강 기법 및 각 증강 기법을 수행하기 위한 증강 모듈들이 존재할 수 있다. 다양한 증강 모듈들에 대해서 도 3에서 상세히 설명된다.
- [0042] <데이터 전처리를 위한 스토리지 장치>
- [0043] 도 2는 일 실시 예에 따른 데이터 전처리를 위한 스토리지 장치(101)의 구성을 나타낸 블록도이다.
- [0044] 도 2를 참조하면, 딥러닝 엔진(110)으로부터 데이터 전처리 파이프라인을 수신하여 데이터를 증강하고, 인공지능 학습을 위한 GPU(190)로 증강된 데이터를 전송하는 스토리지 장치(101)가 개시된다. 스토리지 장치(101)는 FPGA(140)가 탑재된 프로그램가능한(programmable) 저장 장치로, SSD(solid-state drive)일 수 있다.
- [0045] 스토리지 장치(101)는 딥러닝 서버의 저장 장치일 수 있으며, 단순히 학습 데이터를 그대로 GPU(190)(또는, NPU(neural processing unit)(미도시))로 제공하는 것이 아니라, 데이터를 학습 모델의 견고함을 높이는 형태로 가공하여 능동적으로 제공하는 지능형 저장 장치일 수 있다. 일 실시 예에 따르면, 스토리지 장치(101)가 전처리를 수행하는 데이터는 이미지(image), 음성(sound), 비디오(video) 데이터일 수 있다.
- [0046] 일 실시 예에 따르면, 데이터 전처리를 CPU, GPU가 아닌 스토리지 장치(101)가 전담함에 따라, CPU가 다른 딥러닝 작업(예를 들어, 모델 확인(model validation), 추론(inference), 빅데이터 처리(big data processing))을 위해 사용될 수 있고, GPU가 학습 알고리즘의 수행만을 위해 사용될 수 있다.
- [0047] 일 실시 예에 따르면, 스토리지 장치(101)는 고성능 학습을 요구하는 워크로드(workload)에 맞게 FPGA(140) 기반의 구현을 통해 향상된 데이터 전처리를 제공할 수 있다. 예를 들어, FPGA(140)에 재구성가능한(re-configurable) 하나 이상의 증강 모듈(augmentation module)(150)들이 프로그래밍되어 동적인 데이터 증강이 가능할 수 있다. 일 실시 예에 따르면, 스토리지 장치(101)는 SSD 수준에서 전처리 과정을 병렬화(parallelization)하여 전처리 속도를 개선할 수 있다. 일 실시 예에 따르면, 스토리지 장치(101)는 데이터 디코딩의 불필요한 중복 작업(redundancy)을 제거하고 워크로드(workload)를 통합(consolidation)할 수 있다.
- [0048] 일 실시 예에 따르면, 스토리지 장치(101)는 전처리할 로우(raw) 데이터를 저장하는 제1 메모리(133) 및 제1 메모리(133)로부터 데이터를 읽고 제1 메모리(133)에 데이터를 쓰는 것을 관리하는 SSD 컨트롤러(135)를 포함할 수 있다. 제1 메모리는 플래시 메모리(flash memory)일 수 있다.
- [0049] 일 실시 예에 따르면, 스토리지 장치(101)는 FPGA(field programmable gate array)(140)를 포함할 수 있다. FPGA(140)에는 증강 기법을 수행하기 위한 증강 모듈들(150)이 프로그래밍 될 수 있고, 데이터를 디코딩하는 디코더(170), 프로세서(120) 및 디코딩된 데이터를 저장하기 위한 제2 메모리(136)가 포함될 수 있다. 일 실시 예에 따르면, 프로세서는 FPGA용 소프트 마이크로 프로세서 코어(soft microprocessor core) 중 하나인 마이크로블레이즈(microblaze)일 수 있다. 다른 실시 예에 따르면, 프로세서(120)는 RISC-V일 수 있다. 프로세서(120)는 증강 모듈들(150) 간의 데이터 처리, 통신을 관리하고, FPGA(140)의 제3 메모리(139)에 대한 접근을 관리할 수 있다.
- [0050] 일 실시 예에 따르면, 제2 메모리(136)는 BRAM(block RAM)일 수 있다. 일 실시 예에 따르면, 제2 메모리(136)는 데이터의 로딩과 중간 데이터(intermediate data)를 칩 외부로 보내지 않고 내부에서 임시 저장함으로써, 외부 데이터 트래픽을 줄이고 소비 전력을 절감할 수 있다.

- [0051] 일 실시 예에 따르면, 스토리지 장치(101)는 제3 메모리(139)를 더 포함할 수 있다. 일 실시 예에 따르면 제3 메모리는 FPGA(140) 전용 DRAM일 수 있으며, FPGA에서 수행된 중간 데이터들을 임시로 저장할 수 있다. 예를 들어 다양한 증강 모듈들이 사용되는 과정에서 중간 데이터가 생성될 수 있고, 제3 메모리(139)에 임시로 저장될 수 있다. 일 실시 예들에 따르면, 프로세서(120)는 제3 메모리(139)를 활용하여 불필요한 과정을 중복 수행하지 않을 수 있다.
- [0052] 일 실시 예에 따르면, FPGA(140)는 프로세서(120), 다양한 증강 모듈들(150), 제2 메모리(136) 및 제3 메모리(139) 간의 통신을 위한 제1 통신 인터페이스(163)를 포함할 수 있다. 일 실시 예에 따르면, 제1 통신 인터페이스(163)는 병렬, 고성능, 동기식, 고주파수, 다중 마스터, 다중 슬레이브 통신 인터페이스를 지니고, 다중 슬레이브와 마스터 구조를 통해 최적의 프로토콜을 제공하는 AMBA(AXI4)일 수 있다.
- [0053] 일 실시 예에 따르면, FPGA(140)는 제1 메모리(133) 및 GPU(190)와 통신을 위한 제2 통신 인터페이스(167)를 포함할 수 있다. 일 실시 예에 따르면 제2 통신 인터페이스(167)는 PCIe(PCI express)일 수 있다.
- [0054] 일 실시 예에 따르면, 프로세서(120)는 딥러닝 엔진(110)으로부터 데이터 전처리 파이프라인(pipeline)을 수신하고, 이에 기초하여 하나 이상의 증강 모듈들(150) 중 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈을 결정할 수 있다. 다양한 실시 예들에 따른 데이터 전처리 파이프라인에 기초하여 타겟 증강 모듈들을 결정하는 동작은 도 3에서 상세히 설명된다.
- [0055] 일 실시 예에 따르면, 프로세서(120)는 증강 모듈들(150)의 상태(유휴(idle) 또는 동작(busy))를 판단할 수 있고, 타겟 증강 모듈들 중 유휴 증강 모듈을 사용하여 동적으로 제2 메모리(136)에 저장된 디코딩된 데이터를 증강할 수 있다. 디코딩된 데이터들은 증강 모듈들에 의해 병렬 처리될 수 있다. 다양한 실시 예들에 따른 데이터 증강 과정은 도 4 및 도 5에서 상세히 설명된다.
- [0056] 일 실시 예에 따르면, 제2 메모리(136)에 디코딩된 데이터가 존재하지 않는 경우, 프로세서(120)는 제1 메모리(133)에 저장된 로우(raw) 데이터를 페치(fetch)하고, 디코더(170)로 디코딩하여 타겟 증강 모듈들 중 유휴 증강 모듈들로 데이터 증강을 수행할 수 있다. 일 실시 예에 따르면 디코딩된 데이터는 제2 메모리(136)에 저장되어, 추후 동일한 로우 데이터 증강 처리 입력시 디코딩 과정이 반복 수행되지 않을 수 있다.
- [0057] 일 실시 예에 따르면, 프로세서(120)는 제2 메모리(136)에 디코딩된 데이터가 존재하지 않는 경우 제1 메모리(133)로부터 로우(raw) 데이터를 페치(fetch)하기 전 제3 메모리(139)를 참조할 수 있다. 제3 메모리(139)는 FPGA(140) 전용 메모리로 다양한 데이터를 임시 저장하며, 프로세서(120)가 제1 메모리(133)에 접근하는 것 보다 제3 메모리(139)에 접근하는 것이 트래픽을 덜 소모할 수 있다.
- [0058] 일 실시 예에 따르면, 프로세서(120)는 증강된(augmented) 데이터를 GPU(190)로 전송할 수 있다. 증강된 데이터는 제2 통신 인터페이스(167)(예를 들어, PCIe)를 통해 CPU를 거치지 않고 우회(bypass)하여 GPU(190)로 직접 전송될 수 있고, SPIN(seamless operating system integration of peer-to-peer DMA between SSDs and GPUs), GPUDirect Storage 기술이 활용될 수 있다. 일 실시 예에 따르면, 증강된 데이터가 CPU를 거치지 않고 GPU(190)로 전송됨으로써 CPU 리소스 활용이 증대될 수 있다.
- [0060] 도 3은 일 실시 예에 따른 하나 이상의 증강 모듈들 중 타겟 증강 모듈들이 결정되는 동작을 설명하기 위한 도면이다.
- [0061] 도 3을 참조하면, FPGA(140)에 프로그래밍된 증강 모듈들(150)에는 다양한 증강 모듈들이 포함될 수 있고, 예를 들어 픽셀 값은 그대로 유지하면서 배치 구조를 변경함으로써 전체 이미지의 모양을 바꾸는 어파인(affine) 변환을 수행하는 Affine accelerator(151), 이미지 사이즈의 압축과 변환(dnsize)을 수행하는 Pooling accelerator(152), 무작위 연산(예: 무작위 이미지 회전 또는 반전, 무작위 잘라내기(RandomCrop))을 수행하는 Random Number Generator(153), 이미지 정규화(normalize)를 수행하는 MAC Array(154), 이미지의 밝기(brightness), 대비(contrast) 및 채도(saturation)를 무작위로 변경(ColorJitter)하는 RGB-HSV transformer(155)가 포함될 수 있다. 다만 이는 예시적인 증강 모듈이며, 다양한 증강 모듈들이 FPGA(140)에 재구성 가능(re-configurable)하게 프로그래밍될 수 있다.
- [0062] 일 실시 예에 따르면, 프로세서(120)는 딥러닝 엔진(110)으로부터 데이터 전처리를 위한 파이프라인(pipeline)을 수신하고, 하나 이상의 증강 모듈들(151, 152, 153, 154, 155) 중 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들을 결정할 수 있다. 일 실시 예에 따르면, 프로세서(120)는 딥러닝 엔진(110)으로부터 수신한 파이

프라인을 수행하기 위해서 Pooling Accelerator(152), MAC Array(154), RGB-HSV transformer(155)가 필요하다고 결정하고 Pooling Accelerator(152), MAC Array(154) 및 RGB-HSV transformer(155)를 타겟 증강 모듈로 결정할 수 있다. 딥러닝 엔진(110)으로부터 수신한 파이프라인을 그대로 단순 처리하지 않고, 파이프라인을 분석하여 타겟 증강 모듈들을 결정하는 파이프라인 병렬화(pipeline parallelization)를 통해 데이터가 보다 빠르게 증강될 수 있다. 파이프라인 병렬화(pipeline parallelization)에 대한 구체적인 예시는 도 4를 통해 설명된다.

[0063] 증강 모듈들(150)은 이미 동작을 수행하고 있는 경우에는 그 동작이 완료될 때까지 다음 동작을 수행할 수 없기 때문에, 프로세서(120)는 데이터 전처리 파이프라인을 수행할 타겟 증강 모듈들의 상태를 확인할 수 있다. 일 실시 예에 따르면, 프로세서(120)는 증강 모듈들(150)로부터 각 모듈이 유휴 상태(idle)인지, 동작 상태(busy)인지에 대한 정보를 수신할 수 있다. 각 모듈의 상태에 대한 정보는 제2 메모리(136)에 저장될 수 있다.

[0064] 도 3을 참조하면, 프로세서(120)는 타겟 증강 모듈들(152, 154, 155) 중 유휴 증강 모듈들을 사용하여 제2 메모리(136)에 저장된 디코딩된 데이터를 증강할 수 있다. 도 2에서 설명한 바와 같이 제2 메모리(136)에 저장된 디코딩된 데이터가 존재하지 않으면 제1 메모리(133)에 저장된 로우(raw) 데이터를 페치(fetch)하고, 디코더(170)로 디코딩하여 데이터 증강을 수행할 수 있다. 디코딩된 데이터는 제2 메모리(136)에 저장되어, 추후 동일한 로우 데이터 증강 처리 입력시 디코딩 과정이 반복 수행되지 않을 수 있다.

[0065] 구체적인 증강 동작과 관련하여, 스토리지 장치(101)는 다양한 증강 모듈들(150)을 동적으로 구성하고, 데이터의 병렬 처리할 수 있다. 다양한 실시 예에 따른 증강 과정에서의 데이터 병렬 처리에 대해 도 4 및 도 5에서 상세히 설명된다.

[0067] 도 4는 단일 파이프라인에 따른 데이터 전처리 시 병렬 처리로 인한 동작을 설명하기 위한 도면이다.

[0068] 도 4를 참조하면, 딥러닝 엔진(110)으로부터 수신한 하나의 파이프라인에 대해, 파이프라인 병렬화(pipeline parallelization)를 통해 데이터가 병렬 처리되는 동작이 도시된다.

[0069] 일 실시 예에 따르면, 딥러닝 엔진(110)으로부터 수신한 단일 파이프라인에 대해 도 3을 참조하여 설명한 바와 같이 타겟 증강 모듈들이 결정될 수 있고, 처리 순서가 결정될 수 있다. 도 4를 참조하면, 타겟 증강 모듈들 및 처리 순서에 따른 파이프라인(410)이 Random Number Generator(153), MAC array(154) 및 RGB-HSV(155) 순서로 결정될 수 있다.

[0070] 도 4를 참조하면, 딥러닝 엔진(110)으로부터 수신한 파이프라인을 그대로 사용하여 단순 처리하는 경우(430)와 파이프라인 병렬화(pipeline parallelization)를 통해 병렬 처리하는 경우(450)가 비교 설명된다. 스토리지 장치(101)에서 증강할 데이터에는 제1 이미지와 제2 이미지가 포함될 수 있다.

[0071] 일 실시 예에 따르면, 딥러닝 엔진(110)으로부터 수신한 파이프라인을 그대로 사용하여 단순 처리하는 경우(430)에는, 수신한 파이프라인에 따라 제1 이미지를 처리한 후, 제2 이미지를 처리한다. 즉, 타겟 모듈들(153, 154, 155)의 병렬화 없이 제1 이미지 처리 후 제2 이미지가 처리되므로, 도 4에 도시된 바와 같이 제1 이미지가 RandomCrop, Normalize, ColorJitter 처리된 후 제2 이미지가 RandomCrop, Normalize, ColorJitter 처리될 수 있다.

[0072] 일 실시 예에 따르면, 타겟 모듈들(153, 154, 155)이 결정되어 파이프라인 병렬화(pipeline parallelization) 후 데이터를 병렬 처리하는 경우(450)에는, 타겟 모듈들(153, 154, 155)에 따라 제1 이미지 및 제2 이미지가 병렬로 처리될 수 있다. 다시 말하면, 각 타겟 모듈(153, 154, 155) 별로 제1 이미지 처리 후 제2 이미지가 처리될 수 있다.

[0073] 예를 들어 도 4에 도시된 바와 같이 제1 이미지는 RandomCrop, Normalize, ColorJitter 처리되는 경우, 타겟 모듈들(153, 154, 155) 별로 이미지가 처리되므로 Random Number Generator(153)는 제1 이미지의 RandomCrop를 완료하면 유휴(idle) 상태가 되고, 제1 이미지의 전체 증강 과정이 완료되기 전이라도 제2 이미지의 RandomCrop를 수행할 수 있다. 즉 MAC Array(154)에서 제1 이미지의 normalize 과정이 일어나는 동안 Random Number Generator(153)의 제2 이미지의 Randomcrop 과정이 동시에 일어날 수 있다.

[0074] 일 실시 예에 따르면, 각 증강 모듈들에 대한 정보는 프로세서(120)로 전송되고, 프로세서(120)는 이를 활용하여 데이터들을 증강 처리한다. 다만 파이프라인을 수행하기 위한 모든 타겟 모듈이 유휴 상태인 경우 데이터 처리를 시작하는 것은 아니다. 예를 들어 병렬처리(450)에서 제1 이미지 증강 중 제1 이미지를 Normalize할

때, Normalize를 수행하는 MAC array(154)는 동작 상태(busy)이나, Random Number Generator(153) 및 RGB-HSV transformer(155)는 유휴 상태(idle)일 수 있다. 제1 이미지 다음으로 처리할 제2 이미지는 가장 먼저 Random Number Generator(153)에 의해 RandomCrop 되어야 하므로, 비록 타겟 증강 모듈들(153, 154, 155) 중 MAC array(154)가 동작 상태(busy)라 해도 RandomCrop를 수행하는 Random Number Generator(153)는 유휴 상태(idle)이므로 프로세서(120)는 Random Number Generator(153)를 사용하여 제2 이미지를 Randomcrop할 수 있다. 즉 제1 이미지에 대한 전체 증강 과정이 완료되지 않은 상태라 해도 제2 이미지에 대한 증강 처리가 시작될 수 있다.

[0075] 일 실시 예에 따르면, 다양한 타겟 증강 모듈들(153, 154, 155)의 처리에 따른 중간 데이터가 제3 메모리(139)에 저장될 수 있다. 제3 메모리(139)는 FPGA 전용 메모리로, 프로세서(120)는 제1 메모리(133)보다 접근성이 좋은 제3 메모리(139)에 중간 데이터들을 저장하고 활용할 수 있다. 예를 들어 프로세서(120)는 파이프라인(410)에 따라 제1 이미지 증강처리 과정에서 제1 이미지를 Random Number Generator(153)를 통해 처리하고, RandomCrop된 제1 이미지를 제3 메모리(139)에 저장할 수 있다. 추후 프로세서(120)가 다른 파이프라인을 수신하였는데 제1 이미지에 대한 RandomCrop이 필요하다고 판단하면, 다시 Random Number Generator(153) 모듈로 처리하지 않고 제3 메모리(139)에 저장된 RandomCrop된 제1 이미지를 페치(fetch)하여 활용할 수 있다.

[0076] 일 실시 예에 따르면, 파이프라인 병렬화를 통해 데이터 증강 속도가 향상될 수 있으며, 딥러닝 엔진(110)으로부터 수신한 파이프라인이 복잡하고 길수록 병렬 처리로 인한 속도 개선이 증대될 수 있다.

[0078] 도 5는 여러 파이프라인에 따른 데이터 전처리 시 병렬 처리로 인한 동작을 설명하기 위한 도면이다.

[0079] 도 5를 참조하면, 도 4를 참조하여 설명한 단일 파이프라인을 수신하여 이미지를 처리하는 과정에서의 병렬 처리보다 높은 차원의 병렬 처리인 여러 파이프라인에 따른 병렬 처리 동작이 설명된다.

[0080] 도 2를 참조하여 설명한 바와 같이, FPGA 내에 정적으로 구현된 데이터 전처리 모듈들과 파이프라인(pipeline)을 통해 데이터를 전처리하는 방법은 다양한 데이터 증강 방법들을 동적으로 수행할 수 없다. 구체적으로, 도 5에 도시된 바와 같이 Random Number Generator(151), MAC Array(154), RGB-HSV transformer(155) 순서로 처리되는 제1 파이프라인(510) 및 Affine Accelerator(151), Pooling Accelerator(152) 순서로 처리되는 제2 파이프라인(550)이 있는 경우, 정적(static)으로 구현된 FPGA 기반 스토리지 장치는 파이프라인에 맞게 미리 설계되어야 하므로 두 파이프라인을 동시에 수행할 수 없다.

[0081] 일 실시 예에 따른 스토리지 장치(101)에는 다양한 증강 모듈들이 FPGA 내에 프로그래밍되고, 프로세서(120)에 의해 데이터들이 동적으로 처리되므로 정적으로 미리 설계된 FPGA 기반 스토리지 장치와 달리 다양한 파이프라인을 처리할 수 있다.

[0082] 일 실시 예에 따르면, 스토리지 장치(101)의 프로세서(120)는 딥러닝 엔진(110)으로부터 서로 다른 파이프라인을 수신하고, 타겟 모듈을 결정하여 각각 제1 파이프라인(510) 및 제2 파이프라인(550)으로 구성하여 데이터를 병렬 처리할 수 있다. 도 5를 참조하면 제1 파이프라인(510) 및 제2 파이프라인(550)에서 사용되는 타겟 증강 모듈들이 모두 다르므로, 프로세서(120)는 제1 파이프라인(510) 및 제2 파이프라인(550)에 기초하여 이미지들을 각 타겟 모듈들에 따라 증강 처리할 수 있다.

[0083] 도 5에서 제1 파이프라인(510)에서 사용하는 타겟 모듈들(153, 154, 155) 및 제2 파이프라인(550)에서 사용하는 타겟 모듈들(151, 152) 간 중복되는 모듈이 없는 것으로 도시되었으나, 이에 제한되는 것은 아니다. 일 실시 예에 따르면, 딥러닝 엔진(110)으로부터 수신한 여러 파이프라인을 타겟 모듈에 따라 결정한 파이프라인들 간 중복되는 타겟 모듈들이 존재할 수 있고, 예를 들어 어파인(affine) 변환을 수행하는 Affine Accelerator(151)가 여러 파이프라인들 간 중복되는 타겟 모듈일 수 있다. 다만 타겟 모듈이 일부 중복되더라도 해당 타겟 모듈이 유휴 상태이면 동작할 수 있다. 일 실시 예에 따르면, 중복되는 타겟 모듈인 Affine Accelerator(151)는 제1 파이프라인(510)에 따라 이미지를 처리하는 동안 동작 상태(busy)이다가, 제1 파이프라인(510)에 따른 동작 중 어파인 변환을 완료하면 유휴 상태(idle)가 될 수 있다. 도 3에서 전술한 바와 같이 프로세서(120)는 증강 모듈들(150)의 상태에 대한 정보를 수신할 수 있으므로, 프로세서(120)는 유휴 상태가 된 Affine Accelerator(151)로 제2 파이프라인(520)에 따른 동작 중 어파인 변환을 수행할 수 있다.

[0084] 일 실시 예에 따르면, 서로 다른 파이프라인 간 중복되는 증강 과정이 필요한 경우, 각 파이프라인 별로 각각의 타겟 모듈을 이용할 수 있다. 예를 들어 제1 파이프라인(510)과 제2 파이프라인(520)에 모두 어파인 변환이 포함되는 경우, 어파인 변환을 수행하는 Affine Accelerator가 제1 파이프라인(510)이 사용하는 제1 Affine

Accelerator 및 제2 파이프라인(520)이 사용하는 제2 Affine Accelerator로 구분되어 각각 존재할 수 있다.

- [0085] 일 실시 예에 따르면, 동일한 증강 과정을 수행하는 타겟 모듈이 파이프라인에 따라 각각 존재하고, 각 타겟 모듈 사이에 연산량 차이가 나는 경우 프로세서(120)는 파이프라인 간 타겟 모듈을 공유하여 처리할 수 있다. 예를 들어, 제1 파이프라인(510) 및 제2 파이프라인(520)에서 모두 어파인 변환을 수행하고 제1 파이프라인은 제1 Affine Accelerator, 제2 파이프라인은 제2 Affine Accelerator를 사용할 수 있다. 제1 Affine Accelerator는 연산량 과다이고 제2 Affine Accelerator는 유휴상태(idle)인 경우, 프로세서(120)는 제2 Affine Accelerator를 제1 파이프라인(510)의 어파인 변환에 사용할 수 있다.
- [0087] <FPGA 기반 스토리지 장치의 데이터 전처리 방법>
- [0088] 도 6은 일 실시 예에 따른 스토리지 장치의 동작 방법을 설명하기 위한 흐름도이다.
- [0089] 동작 610 내지 650은 도 2를 참조하여 전술된 FPGA 기반 스토리지 장치(101)의 프로세서(120)에 의해 수행될 수 있고, 간명한 설명을 위해 도 1 내지 도 5를 참조하여 설명한 내용과 중복되는 내용은 생략될 수 있다.
- [0090] 일 실시 예에 따르면 동작 610에서, 프로세서(120)는 인공지능 엔진(110)으로부터 데이터 전처리 파이프라인을 수신할 수 있다. 데이터 전처리 파이프라인에는 스토리지 장치(101)의 제1 메모리(133)에 저장된 로우(raw) 데이터를 어떤 증강 모듈을 어떤 순서로 사용하여 증강할 것인지에 대한 정보가 포함되어 있을 수 있다.
- [0091] 일 실시 예에 따르면 동작 620에서, 프로세서(120)는 데이터 전처리 파이프라인에 기초하여 유휴 증강 모듈을 선택할 수 있다. 데이터 전처리를 위한 FPGA 기반 스토리지 장치(101)의 FPGA(140)에는 재구성가능한(re-configurable) 다양한 증강 모듈들(150)이 프로그래밍 되어 있을 수 있고, 도 3을 참조하여 설명한 바와 같이 프로세서(120)는 파이프라인에 기초하여 로우(raw) 데이터를 증강하기 위한 타겟 증강 모듈들을 결정할 수 있다. 다양한 실시 예들에 따른 유휴 증강 모듈이 선택되는 과정은 도 7에서 상세히 설명된다.
- [0092] 일 실시 예에 따르면 동작 630에서 프로세서(120)는 제2 메모리로부터 디코딩된 데이터를 페치(fetch)할 수 있다. 제2 메모리에 디코딩된 데이터가 존재하지 않는 경우, 프로세서(120)는 먼저 제3 메모리(139)를 참조하여 로우(raw) 데이터가 존재하는지 확인하고, 제3 메모리(139)에도 로우 데이터가 존재하지 않는 경우 제1 메모리(133)로부터 로우 데이터를 페치하여 디코더(170)로 디코딩을 수행할 수 있다. 다양한 실시 예들에 따른 디코딩된 데이터가 페치되는 동작은 도 8에서 상세히 설명된다.
- [0093] 일 실시 예에 따르면 동작 640에서 프로세서(120)는 유휴 증강 모듈들을 사용하여 디코딩된 데이터를 증강할 수 있다. 일 실시 예에 따르면, 동작 620에서 선택된 유휴 증강 모듈들이 사용될 수 있고, 프로세서(120)는 각 증강 모듈이 증강 처리할 때마다 중간 데이터를 제3 메모리(139)에 임시로 저장할 수 있다. 도 4에서 설명한 바와 같이, 프로세서(120)가 제3 메모리(139)에 중간 데이터를 저장함으로써 데이터 전처리 효율이 증대될 수 있다.
- [0094] 일 실시 예에 따르면 동작 650에서 프로세서(120)는 증강된 데이터를 GPU(190)로 전송할 수 있다. 일 실시 예에 따르면, 도 2에서 설명한 바와 같이 증강된 데이터가 CPU를 거치지 않고 GPU(190)로 전송될 수 있고, CPU 리소스 활용이 증대될 수 있다.
- [0096] 도 7은 일 실시 예에 따른 데이터 전처리 파이프라인에 기초하여 유휴 증강 모듈들을 선택하는 동작을 설명하기 위한 흐름도이다.
- [0097] 동작 710 내지 730은 도 2를 참조하여 전술된 FPGA 기반 스토리지 장치(101)의 프로세서(120)에 의해 수행될 수 있다. 일 실시 예에 따르면, 동작 710 내지 730은 도 6을 참조하여 설명한 데이터 전처리 파이프라인에 기초하여 유휴 증강 모듈들을 선택하는 동작(예: 도 6의 동작 620)에 대응할 수 있다.
- [0098] 일 실시 예에 따르면 동작 710에서, 프로세서(120)는 파이프라인을 타겟 증강 모듈에 따라 파이프라인을 병렬화(pipeline parallelization)할 수 있다. 도 3 내지 도 5를 참조하여 전술한 바와 같이, 프로세서(120)는 딥러닝 엔진(110)으로부터 수신한 데이터 전처리 파이프라인을 분석하여 파이프라인을 수행하기 위해 필요한 타겟 증강 모듈들을 결정하고, 증강 모듈들의 처리 순서를 결정할 수 있다.
- [0099] 일 실시 예에 따르면 동작 720에서, 프로세서(120)는 타겟 증강 모듈들의 유휴 상태에 대한 정보를 수신할 수

있다. 도 2에서 전술한 바와 같이 프로세서(120)는 제1 통신 인터페이스(163)를 통해 여러 증강 모듈들의 상태에 대한 정보를 수신할 수 있다.

[0100] 일 실시 예에 따르면 동작 730에서, 프로세서(120)는 타겟 유희 모듈들 중 유희 증강 모듈을 선택할 수 있다. 도 7을 참조하면 동작 730은 도 6의 동작 620에 속하는 것으로, 선택된 유희 증강 모듈들을 사용하여 디코딩된 데이터를 증강하는 동작(예: 도 6의 동작 640) 이전에 수행되는 것으로 도시되었지만, 이에 제한되는 것은 아니다. 즉, 각 모듈들에 대한 상태 정보가 프로세서(120)로 전송되고, 프로세서(120)는 파이프라인에 기초하여 결정된 타겟 증강 모듈들 및 처리 순서에 따라 디코딩된 데이터를 증강 처리함에 있어 유희상태인 증강 모듈을 사용하는 것일 뿐, 도 4를 참조하여 설명한 바와 같이 타겟 모듈들이 모두 유희 상태일 때 증강 처리를 시작하는 것은 아니다.

[0102] 도 8은 일 실시 예에 따른 디코딩된 데이터를 페치(fetch)하는 동작을 설명하기 위한 흐름도이다.

[0103] 동작 810 내지 830은 도 2를 참조하여 전술된 FPGA 기반 스토리지 장치(101)의 프로세서(120)에 의해 수행될 수 있다. 일 실시 예에 따르면, 동작 810 내지 830은 도 6을 참조하여 설명한 제2 메모리로부터 디코딩된 데이터를 페치(fetch)하는 동작(예: 도 6의 동작 630)에 대응할 수 있다.

[0104] 일 실시 예에 따르면 동작 810에서 프로세서(120)는 제2 메모리(136)를 참조하여 디코딩된 데이터가 있는지 확인할 수 있다.

[0105] 일 실시 예에 따르면, 제2 메모리(136)에 디코딩된 데이터가 없는 경우, 동작 820에서 프로세서는 제1 메모리(133)에 저장된 로우(raw) 데이터를 디코딩하고, 제2 메모리(136)에 저장할 수 있다. 도 8에서는 간명한 설명을 위해 동작 820에서 로우 데이터를 디코딩하여 제2 메모리(136)에 저장하고 동작 830에서 제2 메모리(136)에 저장된 디코딩된 데이터를 페치(fetch)하는 것으로 도시되었지만, 이에 제한되지 않는다. 일 실시 예에 따르면 프로세서(120)는 제2 메모리(136)에 저장하기 전 디코딩된 데이터를 페치(fetch)하여 동작 640에서 증강 처리할 수 있다.

[0106] 일 실시 예에 따르면, 도 2에서 전술한 바와 같이, 프로세서(120)는 동작 810에서 제2 메모리(136)에 디코딩된 데이터가 존재하지 않는 경우 제1 메모리(133)로부터 로우 데이터를 가져오기 전 제3 메모리(139)에 로우 데이터가 존재하는지 확인할 수 있다. 제3 메모리(139)는 FPGA(140) 전용 메모리로, 프로세서(120)가 제1 메모리(133)에 접근하는 것 보다 제3 메모리(139)에 접근하는 것이 트래픽을 덜 소모할 수 있다.

[0107] 일 실시 예에 따르면 제2 메모리(136)에 디코딩된 데이터가 있는 경우, 동작 830에서 프로세서(120)는 제2 메모리(136)에 저장된 디코딩된 데이터를 페치(fetch)할 수 있다. 동작 810 내지 동작 830을 통해 페치(fetch)된 디코딩된 데이터는 증강 처리되고, 인공지능 학습을 위해 GPU(190)로 전송될 수 있다.

[0109] 이상에서 설명된 실시예들은 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치, 방법 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(field programmable gate array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 컨트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0110] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상

장치(virtual equipment), 컴퓨터 저장 매체 또는 장치, 또는 전송되는 신호 파(signal wave)에 영구적으로, 또는 일시적으로 구체화(embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

[0111] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 저장할 수 있으며 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.

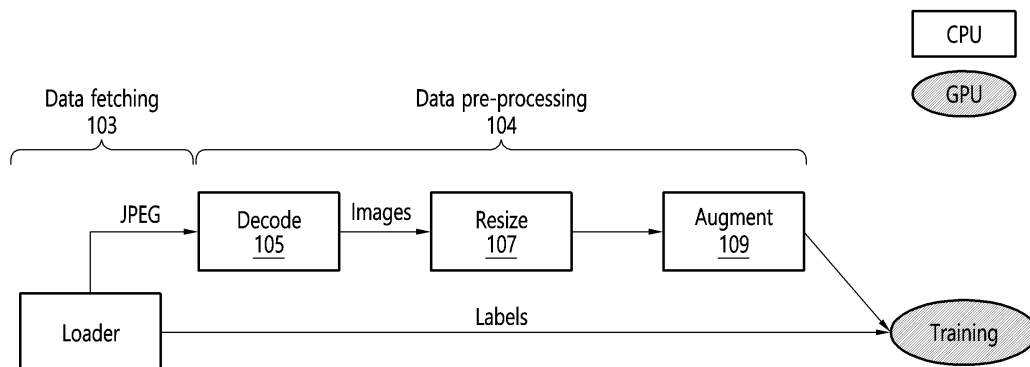
[0112] 위에서 설명한 하드웨어 장치는 실시예의 동작을 수행하기 위해 하나 또는 복수의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

[0113] 이상과 같이 실시예들이 비록 한정된 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 이를 기초로 다양한 기술적 수정 및 변형을 적용할 수 있다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

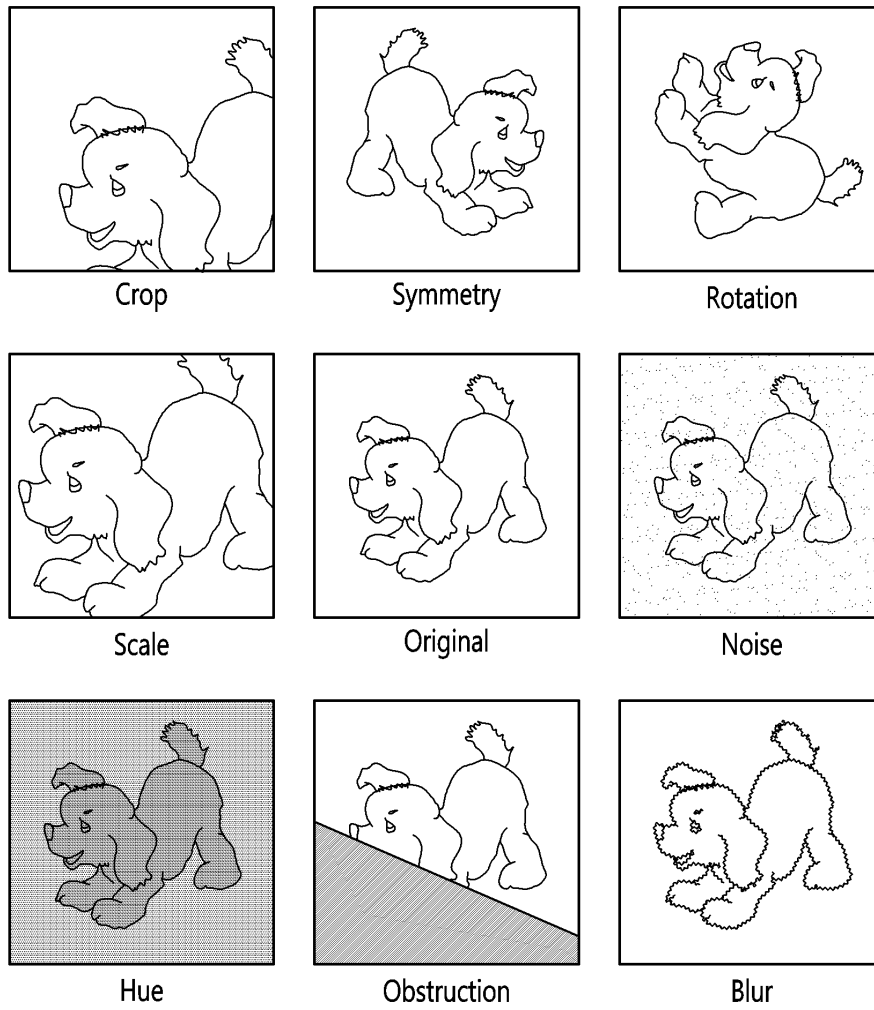
[0114] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

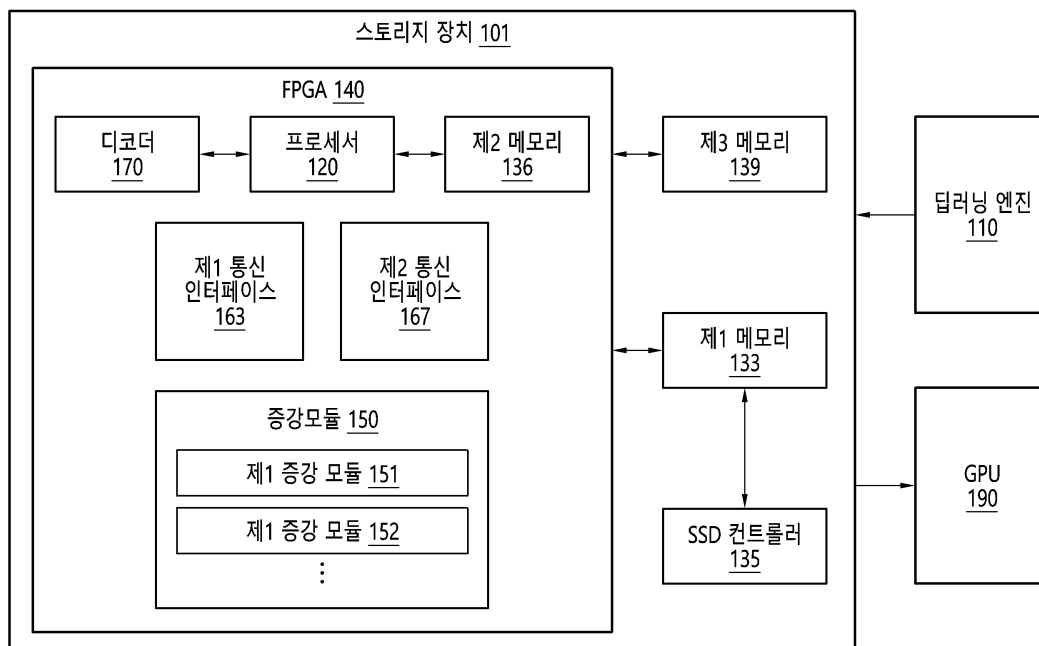
도면1a



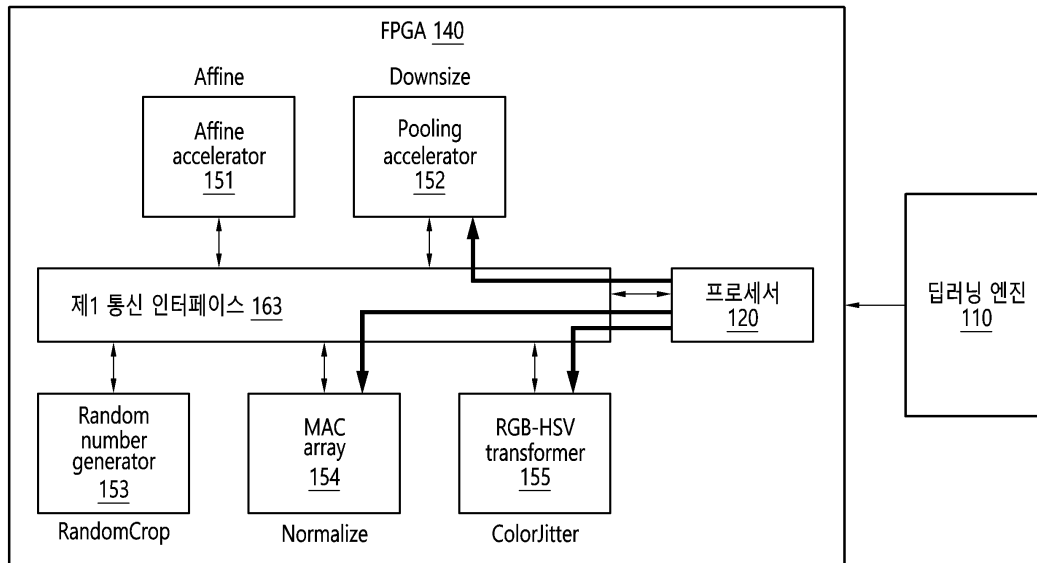
도면1b



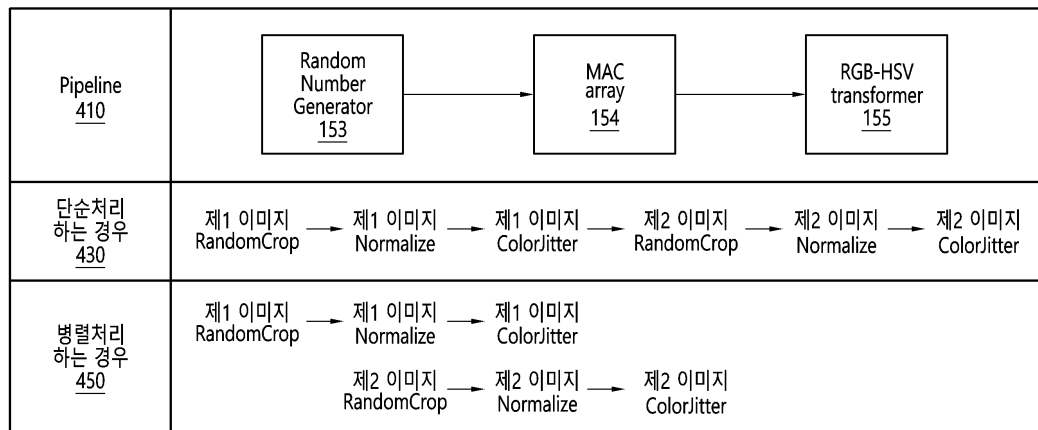
도면2



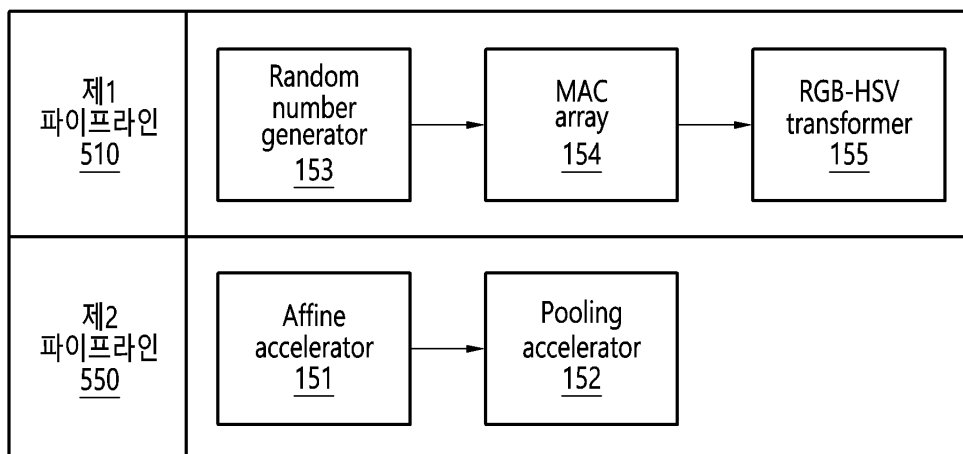
도면3



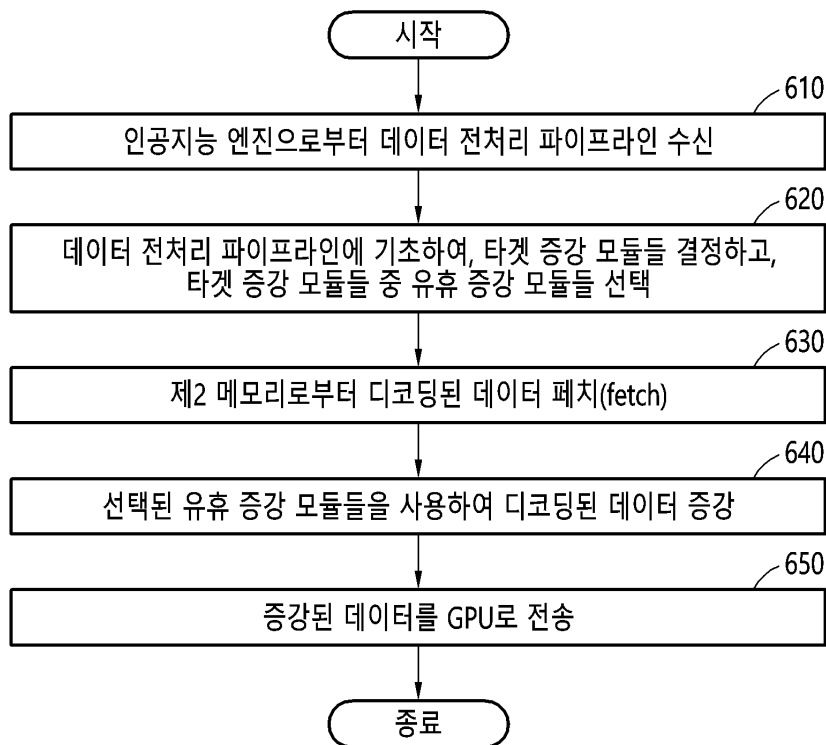
도면4



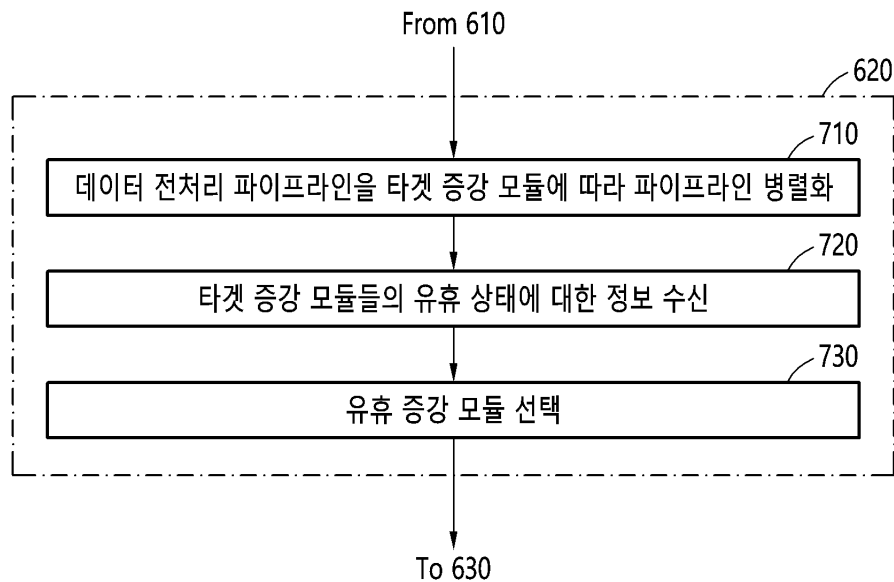
도면5



도면6



도면7



도면8

