(12) 특허협력조약에 의하여 공개된 국제출원

(19) 세계지식재산권기구
국제사무국

(43) 국제공개일
2020 년 9 월 17 일 (17.09.2020)

WIPO | PCT

(10) 국제공개번호
WO 2020/184827 A1

(51) 국제특허분류:
G06F 9/50 (2006.01) *G06F 9/445* (2006.01)
G06F 9/48 (2006.01)

(21) 국제출원번호: PCT/KR2020/000230

(22) 국제출원일: 2020 년 1 월 7 일 (07.01.2020)

(25) 출원언어: 한국어

(26) 공개언어: 한국어

(30) 우선권정보:
10-2019-0029520 2019 년 3 월 14 일 (14.03.2019) KR

(71) 출원인: 삼성전자주식회사 (**SAMSUNG ELECTRONICS CO., LTD.**) [KR/KR]; 16677 경기도 수원시 영통구 삼성로 129, Gyeonggi-do (KR).

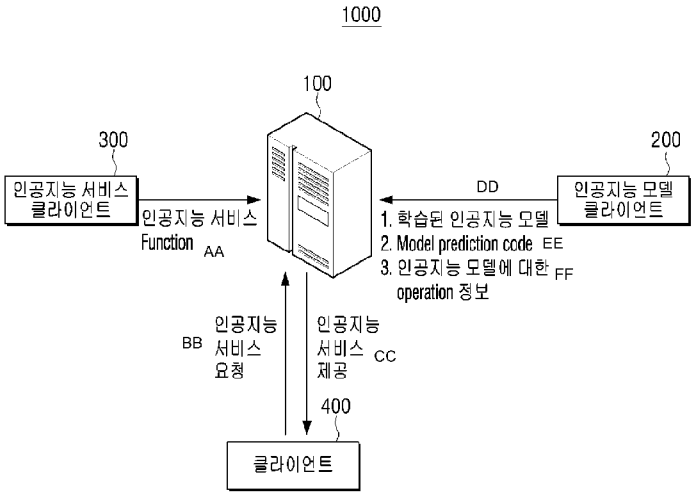
(72) 발명자: 김근섭 (**KIM, Keunseob**); 16677 경기도 수원시 영통구 삼성로 129, Gyeonggi-do (KR). 황정동 (**HWANG, Jeongdong**); 16677 경기도 수원시 영통구 삼성로 129, Gyeonggi-do (KR). 박재만 (**PARK, Jaeman**); 16677 경기도 수원시 영통구 삼성로 129, Gyeonggi-do (KR).

(74) 대리인: 김태헌 등 (**KIM, Tae-hun** et al.); 06626 서울시 서초구 강남대로343 신덕빌딩 9층, Seoul (KR).

(81) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 국내 권리의 보호를 위하여): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD,

(54) **Title:** ELECTRONIC DEVICE AND METHOD OF CONTROLLING SAME

(54) **발명의 명칭:** 전자 장치 및 이의 제어 방법



200 ... Artificial intelligence model client
300 ... Artificial intelligence service client
400 ... Client
AA ... Artificial intelligence service function
BB ... Request artificial intelligence service
CC ... Provide artificial intelligence service
DD ... 1. Trained artificial intelligence model
EE ... 2. Model prediction code
FF ... 3. Operation information on artificial intelligence model

(57) **Abstract:** Disclosed is a method for providing an artificial intelligence service based on a serverless platform. The method comprises the steps of: determining a container into which artificial intelligence models are to be loaded, on the basis of features of a plurality of containers and features of a plurality of artificial intelligence models registered in a model store; loading the artificial intelligence models into the container; upon receiving a request for an artificial intelligence service from a client, obtaining a function corresponding to the requested artificial intelligence service from a database; determining the container in which the artificial intelligence model corresponding to the requested artificial intelligence service is loaded; by executing the obtained function on the container, acquiring data with regard to the request, from the artificial intelligence model loaded in the container; and transferring the acquired data to a client.

WO 2020/184827 A1

[다음 쪽 계속]

SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 역내 권리의 보호를 위하여): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 유라시아 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 유럽 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

공개:

— 국제조사보고서와 함께 (조약 제21조(3))

(57) 요약서: 서버리스(serverless) 플랫폼에 기반한 인공지능 서비스 제공 방법이 개시된다. 본 제공 방법은, 복수의 컨테이너의 속성 및 모델 스토어에 등록된 복수의 인공지능 모델의 속성에 기초하여 인공지능 모델이 로딩될 컨테이너를 판단하는 단계, 컨테이너에 인공지능 모델을 로딩하는 단계, 클라이언트로부터 인공지능 서비스에 대한 요청이 수신되면, 데이터베이스로부터 요청된 인공지능 서비스에 대응되는 함수를 획득하는 단계, 요청된 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단하는 단계, 컨테이너에서 획득된 함수를 실행하여 컨테이너에 로딩된 인공지능 모델로부터 요청에 대한 데이터를 획득하는 단계 및 획득된 데이터를 클라이언트에게 전송하는 단계를 포함한다.

명세서

발명의 명칭: 전자 장치 및 이의 제어 방법

기술분야

[1] 본 개시는 전자 장치 및 이의 제어방법에 관한 것으로, 더욱 상세하게는 서버리스 기반 시스템에서 인공지능 서비스를 제공하는 전자 장치 및 이의 제어방법에 대한 것이다.

배경기술

[2] 최근 클라우드 컴퓨팅(cloud computing) 기술이 발달하면서, 컨테이너(Container) 기반 컴퓨팅 기술 및 서버리스(serverless)기반 컴퓨팅 기술에 대한 관심이 증가하고 있다.

[3] 특히, 요즘은 컨테이너 기반 컴퓨팅 기술과 인공지능 기술이 결합되어, 컨테이너 기반 컴퓨팅 환경에서 인공지능 서비스(Artificial Intelligence Service, AI Service)를 제공하는 기술이 널리 사용되고 있다.

[4] 이와 관련하여, 도 1은 종래의 컨테이너 기반 컴퓨팅 시스템에서 제공되는 인공지능 서비스를 설명하기 위한 도면이다.

[5] 컨테이너 기반 환경에서 인공지능 모델 클라이언트(또는 인공지능 모델 개발자)는 학습된 인공지능 모델 파일, 학습된 모델에 대한 model prediction code 및 인공지능 모델에 대한 operation 정보를 인공지능 서비스 클라이언트(또는 인공지능 서비스 개발자)에게 제공할 수 있다. 여기에서, 학습된 인공지능 모델 파일은 학습된 인공지능 모델의 히든 레이어(hidden layer) 및 인공 지능 모델이 결과 값으로 출력하는 레이블링(labeling) 정보를 포함한 파일이고, 학습된 모델에 대한 model prediction code는 학습된 모델의 입력 값에 대한 결과 값을 얻기 위해 필요한 코드 또는 프로그램을 포함하며, 인공지능 모델에 대한 operation 정보는 인공지능 모델을 이용하는데 최소한으로 필요한 CPU, GPU, 메모리 등의 리소스(resource) 정보를 포함할 수 있다.

[6] 컨테이너 기반 환경에서 인공지능 서비스 클라이언트는 인공지능 모델 클라이언트로부터 획득된 학습된 인공지능 모델 파일 및 학습된 모델에 대한 prediction code를 반영하여 인공지능 서비스를 제공하는 함수를 생성하고, 생성된 함수를 인공지능 서비스 서버(Artificial Intelligence Server, AI Server)에 제공할 수 있다.

[7] 그리고, 인공지능 서비스를 이용하려는 클라이언트는 인공지능 서비스 서버에 인공지능 서비스를 요청하고, 서버는 서비스 클라이언트에 의해 제공된 함수를 실행하여 획득한 결과 값을 클라이언트에게 제공할 수 있다.

[8] 한편, 컨테이너 기반 환경에서 인공지능 서비스 클라이언트는 인공지능 서비스를 제공하기 위한 컨테이너를 생성하고, 이를 인공지능 서비스 서버(Artificial Intelligence Server, AI Server)에 제공할 수 있다.

[9] 인공지능 서비스 클라이언트는 컨테이너 내에 학습된 인공지능 모델 파일(또는 모델 바이너리(binary)) 및 REST(Representational State Transfer)ful API Server를 구현할 수 있다. 구체적으로, 인공지능 서비스 클라이언트는 컨테이너 내에 컨테이너의 Auto scaling, 고가용성(High Availability) 등을 보장하는 RESTful API Server 서버를 생성하고 이를 관리할 수 있다.

[10] 한편, 컨테이너 기반 인공지능 서비스 제공 환경에서는 컨테이너의 고가용성을 위하여, 즉, 컨테이너가 항상 유지될 수 있도록 하기 위하여, 하나의 인공지능 서비스에 대한 복수의 컨테이너가 제공될 수 있다.

[11] 인공지능 서비스 클라이언트에 의해 제공된 컨테이너는 인공지능 서비스 서버의 CPU 또는 GPU를 이용하여 인공지능 서비스를 클라이언트에 제공할 수 있다. 이를 위하여, 인공지능 서비스 클라이언트는 컨테이너의 유지 및 관리에 필요한 인공지능 서비스 서버의 CPU, GPU, 메모리 등의 리소스를 컨테이너에 할당(provisioning)할 수 있다.

[12] CPU를 이용하여 인공지능 서비스가 제공되는 경우, 인공지능 서비스의 수행 속도 등 효율이 낮다는 단점이 있으며, GPU를 이용하여 인공지능 서비스가 제공되는 경우, GPU 가격이 비싸다는 점에서 비용이 많이 든다는 단점이 있다.

[13] 이에 따라, 컨테이너 기반 환경에서 인공지능 서비스를 제공하는 대신, 서버리스(serverless)환경에서 인공지능 서비스를 제공하려는 시도가 나타나고 있다.

[14] 서버리스 컴퓨팅은 개발자가 서버의 프로비저닝이나 운영을 신경 쓸 필요 없이, 어플리케이션의 소스 코드(또는 함수(function))만 작성해서 서버에 등록하면 플랫폼이 알아서 해당 코드를 실행해주고 결과를 반환해주는 시스템이다.

[15] 서버리스란 단어 자체로 보면 서버가 없다고 생각할 수 있지만 물리적으로 서버가 존재하지 않다는 것을 의미하지는 않는다. 서버리스 컴퓨팅은 이벤트 기반으로 필요할 때만 리소스를 사용하는 방식을 제공하는 것으로서, 해당 이벤트에 대해 별도로 할당된 전용 서버가 없다는 의미에서 서버리스라고 부른다.

발명의 상세한 설명

기술적 과제

[16] 본 개시의 목적은 서버리스 컴퓨팅 시스템 환경에서 인공지능 서비스를 제공하는 전자 장치 및 이의 제어방법을 제공함에 있다.

과제 해결 수단

[17] 본 개시의 일 실시 예에 따른 서버리스(serverless) 플랫폼에 기반한 인공지능 서비스 제공 방법은, 인공지능 모델을 위한 라이브러리가 로딩된 복수의 컨테이너의 속성 및 모델 스토어에 등록된 복수의 인공지능 모델의 속성에 기초하여, 인공지능 모델이 로딩될 컨테이너를 판단하는 단계, 상기 컨테이너에

로딩된 라이브러리를 바탕으로 상기 컨테이너에 상기 인공지능 모델을 로딩하는 단계, 클라이언트로부터 인공지능 서비스에 대한 요청이 수신되면, 복수의 함수를 포함하는 데이터베이스로부터 상기 요청된 인공지능 서비스에 대응되는 함수를 획득하는 단계, 인공지능 모델이 로딩된 상기 복수의 컨테이너 중에서 상기 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단하는 단계, 상기 컨테이너에서 상기 컨테이너에 로딩된 라이브러리를 바탕으로 상기 획득된 함수를 실행하여, 상기 판단된 컨테이너에 로딩된 인공지능 모델로부터 상기 요청에 대한 데이터를 획득하는 단계; 및 상기 획득된 데이터를 상기 클라이언트로 전송하는 단계를 포함할 수 있다.

[18] 그리고, 상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는, 상기 복수의 컨테이너에 할당된 리소스 및 상기 복수의 인공지능 모델에 요구되는 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단하는 단계를 포함할 수 있다.

[19] 또한, 상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는, 상기 복수의 컨테이너 각각에 할당된 리소스 및 상기 각 컨테이너에 로딩된 인공지능 모델에 요구되는 리소스에 기초하여 상기 각 컨테이너에서 사용 가능한 리소스를 판단하고, 상기 판단된 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단하는 단계를 포함할 수 있다.

[20] 그리고, 상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는, 상기 모델 스토어에 인공지능 모델이 등록되는 경우, 상기 복수의 컨테이너의 속성 및 상기 등록된 인공지능 모델의 속성에 기초하여 상기 등록된 인공지능 모델이 로딩될 컨테이너를 판단하는 단계를 포함할 수 있다.

[21] 또한, 상기 컨테이너가 제1 상태인 상태에서 상기 획득된 함수를 실행하면, 상기 컨테이너의 상태를 제2 상태인 것으로 판단하는 단계; 및

[22] 상기 복수의 컨테이너 중 제1 상태인 컨테이너의 개수가 기 설정된 개수 미만이면, 새로운 컨테이너를 생성하는 단계;를 더 포함할 수 있다.

[23] 그리고, 상기 획득된 함수를 실행한 컨테이너가 기설정된 시간 동안 상기 컨테이너에 포함된 모델을 이용하는 인공지능 서비스에 대한 요청을 수신하지 않은 경우, 상기 획득된 함수를 실행한 컨테이너를 킬(kill)하는 단계;를 더 포함할 수 있다.

[24] 여기에서, 상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는, 상기 킬 된 컨테이너를 제외한 나머지 컨테이너의 속성 및 상기 킬 된 컨테이너에 로딩되어 있던 인공지능 모델의 속성에 기초하여 상기 킬 된 컨테이너에 로딩되어 있던 인공지능 모델이 새롭게 로딩될 컨테이너를 판단하는 단계를 포함할 수 있다.

[25] 그리고, 상기 복수의 컨테이너는, GPU(Graphic Processing Unit) 또는 CPU(Central Processing Unit)를 기반으로 상기 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수를 실행할 수 있다.

[26] 또한, 각 컨테이너에서의 상기 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수의 실행 시간에 대한 정보를 수집하는 단계;를 더 포함하고, 상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는, 상기 수집된 정보를 바탕으로 상기 GPU 기반의 컨테이너 및 상기 CPU 기반의 컨테이너 중에서 상기 함수가 실행될 컨테이너를 판단하는 단계를 포함할 수 있다.

[27] 한편, 본 개시의 일 실시 예에 따른 서버리스(serverless) 플랫폼에 기반하여 인공지능 서비스를 제공하는 전자 장치는 통신부, 복수의 함수를 포함하는 데이터베이스를 포함하는 메모리, 및 인공지능 모델을 위한 라이브러리가 로딩된 복수의 컨테이너의 속성 및 모델 스토어에 등록된 복수의 인공지능 모델의 속성에 기초하여, 인공지능 모델이 로딩될 컨테이너를 판단하고, 상기 컨테이너에 로딩된 라이브러리를 바탕으로 상기 컨테이너에 상기 인공지능 모델을 로딩하고, 상기 통신부를 통해 클라이언트 장치로부터 인공지능 서비스에 대한 요청이 수신되면, 상기 데이터베이스로부터 상기 요청된 인공지능 서비스에 대응되는 함수를 획득하고, 상기 복수의 컨테이너 중에서 상기 요청된 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단하고, 상기 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너에서 상기 컨테이너에 로딩된 라이브러리를 바탕으로 상기 획득된 함수를 실행하여, 상기 판단된 컨테이너에 로딩된 인공지능 모델로부터 상기 요청에 대한 데이터를 획득하고, 상기 획득된 데이터를 상기 통신부를 통해 상기 클라이언트로 전송하는 프로세서를 포함할 수 있다.

[28] 그리고, 상기 프로세서는, 상기 복수의 컨테이너에 할당된 리소스 및 상기 복수의 인공지능 모델에 요구되는 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단할 수 있다.

[29] 또한, 상기 프로세서는, 상기 복수의 컨테이너 각각에 할당된 리소스 및 상기 각 컨테이너에 로딩된 인공지능 모델에 요구되는 리소스에 기초하여 상기 각 컨테이너에서 사용 가능한 리소스를 판단하고, 상기 판단된 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단할 수 있다.

[30] 그리고, 상기 프로세서는, 상기 모델 스토어에 인공지능 모델이 등록되는 경우, 상기 복수의 컨테이너의 속성 및 상기 등록된 인공지능 모델의 속성에 기초하여 상기 등록된 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다.

[31] 또한, 상기 프로세서는, 상기 컨테이너가 제1 상태인 상태에서 상기 획득된 함수를 실행하면, 상기 컨테이너의 상태가 제2 상태인 것으로 판단하고, 상기 복수의 컨테이너 중 제1 상태인 컨테이너의 개수가 기 설정된 수 미만이면, 새로운 컨테이너를 생성할 수 있다.

[32] 그리고, 상기 프로세서는, 상기 획득된 함수를 실행한 컨테이너가 기 설정된 시간 동안 상기 컨테이너에 포함된 모델을 이용하는 인공지능 서비스에 대한 요청을 수신하지 않은 경우, 상기 획득된 함수를 실행한 컨테이너를 킬(kill) 할 수 있다.

[33] 또한, 상기 프로세서는, 상기 킬 된 컨테이너를 제외한 나머지 컨테이너의 속성 및 상기 킬 된 컨테이너에 로딩되어 있던 인공지능 모델의 속성에 기초하여 상기 킬 된 컨테이너에 로딩되어 있던 인공지능 모델이 새롭게 로딩될 컨테이너를 판단할 수 있다.

[34] 그리고, 상기 복수의 컨테이너는, GPU(Graphic Processing Unit) 또는 CPU(Central Processing Unit)를 기반으로 상기 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수를 실행할 수 있다.

[35] 또한, 상기 프로세서는, 각 컨테이너에서의 상기 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수의 실행 시간에 대한 정보를 수집하고, 상기 수집된 정보를 바탕으로 상기 함수가 실행될 컨테이너를 판단할 수 있다.

발명의 효과

[36] 본 개시에 따른 전자 장치는 서버리스 컴퓨팅 시스템 환경에서 리소스를 효율적으로 이용하면서 인공지능 서비스를 제공할 수 있다.

도면의 간단한 설명

[37] 도 1은 종래의 컨테이너 컴퓨팅 시스템을 설명하기 위한 도면,

[38] 도 2는 본 개시의 전자 장치를 포함하는 서버리스 컴퓨팅 시스템을 설명하기 위한 도면,

[39] 도 3은 본 개시의 일 실시 예에 따른 전자 장치의 구성을 설명하기 위한 블록도,

[40] 도 4는 본 개시의 일 실시 예에 따른 컨테이너를 설명하기 위한 도면,

[41] 도 5는 본 개시의 일 실시 예에 따라 클라이언트의 요청에 대응되는 인공지능 서비스를 제공하는 전자 장치를 설명하기 위한 도면, 및

[42] 도 6은 본 개시의 일 실시 예에 따른 전자 장치를 제어하는 방법을 설명하기 위한 흐름도이다.

발명의 실시를 위한 최선의 형태

[43] 이하, 본 개시의 다양한 실시 예가 기재된다. 그러나, 이는 본 개시의 기술을 특정한 실시 형태에 대해 한정하려는 것이 아니며, 본 개시의 실시 예들의 다양한 변경(modifications), 균등물(equivalents), 및/또는 대체물(alternatives)을 포함하는 것으로 이해되어야 한다.

[44] 본 문서에서, "가진다," "가질 수 있다," "포함한다," 또는 "포함할 수 있다" 등의 표현은 해당 특징(예: 수치, 기능, 동작, 또는 부품 등의 구성요소)의 존재를 가리키며, 추가적인 특징의 존재를 배제하지 않는다.

[45] 본 문서에서, "A 또는 B," "A 또는/및 B 중 적어도 하나," 또는 "A 또는/및 B 중 하나 또는 그 이상"등의 표현은 함께 나열된 항목들의 모든 가능한 조합을 포함할 수 있다. 예를 들면, "A 또는 B," "A 및 B 중 적어도 하나," 또는 "A 또는 B 중 적어도 하나"는, (1) 적어도 하나의 A를 포함, (2) 적어도 하나의 B를 포함, 또는 (3) 적어도 하나의 A 및 적어도 하나의 B 모두를 포함하는 경우를 모두 지칭할 수 있다.

[46] 본 문서에서 사용된 "제1," "제2," "첫째," 또는 "둘째," 등의 표현들은 다양한 구성요소들을, 순서 및/또는 중요도에 상관없이 수식할 수 있고, 한 구성요소를 다른 구성요소와 구분하기 위해 사용될 뿐 해당 구성요소들을 한정하지 않는다. 예를 들면, 제1 사용자 기기와 제2 사용자 기기는, 순서 또는 중요도와 무관하게, 서로 다른 사용자 기기를 나타낼 수 있다. 예를 들면, 본 문서에 기재된 권리 범위를 벗어나지 않으면서 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소도 제1 구성요소로 바꾸어 명명될 수 있다.

[47] 본 문서에서 사용된 "모듈", "유닛", "부(part)" 등과 같은 용어는 적어도 하나의 기능이나 동작을 수행하는 구성요소를 지칭하기 위한 용어이며, 이러한 구성요소는 하드웨어 또는 소프트웨어로 구현되거나 하드웨어 및 소프트웨어의 결합으로 구현될 수 있다. 또한, 복수의 "모듈", "유닛", "부(part)" 등은 각각이 개별적인 특정한 하드웨어로 구현될 필요가 있는 경우를 제외하고는, 적어도 하나의 모듈이나 칩으로 일체화되어 적어도 하나의 프로세서로 구현될 수 있다.

[48] 어떤 구성요소(예: 제1 구성요소)가 다른 구성요소(예: 제2 구성요소)에 "(기능적으로 또는 통신적으로) 연결되어((operatively or communicatively) coupled with/to)" 있다거나 "접속되어(connected to)" 있다고 언급된 때에는, 상기 어떤 구성요소가 상기 다른 구성요소에 직접적으로 연결되거나, 다른 구성요소(예: 제 3 구성요소)를 통하여 연결될 수 있다고 이해되어야 할 것이다. 반면에, 어떤 구성요소(예: 제1 구성요소)가 다른 구성요소(예: 제2 구성요소)에 "직접 연결되어" 있다거나 "직접 접속되어" 있다고 언급된 때에는, 상기 어떤 구성요소와 상기 다른 구성요소 사이에 다른 구성요소(예: 제3 구성요소)가 존재하지 않는 것으로 이해될 수 있다.

[49] 본 문서에서 사용된 표현 "~하도록 구성된(또는 설정된)(configured to)"은 상황에 따라, 예를 들면, "~에 적합한(suitable for)," "~하는 능력을 가지는(having the capacity to)," "~하도록 설계된(designed to)," "~하도록 변경된(adapted to)," "~하도록 만들어진(made to)," 또는 "~를 할 수 있는(capable of)"과 바꾸어 사용될 수 있다. 용어 "~하도록 구성된(또는 설정된)"은 하드웨어적으로 "특별히 설계된(specifically designed to)" 것만을 반드시 의미하지 않을 수 있다. 대신, 어떤 상황에서는, "~하도록 구성된 장치"라는 표현은, 그 장치가 다른 장치 또는 부품들과 함께 "~할 수 있는" 것을 의미할 수 있다. 예를 들면, 문구 "A, B, 및 C를 수행하도록 구성된(또는 설정된) 프로세서"는 해당 동작을 수행하기 위한 전용 프로세서(예: 임베디드 프로세서), 또는 메모리 장치에 저장된 하나 이상의 소프트웨어 프로그램들을 실행함으로써, 해당 동작들을 수행할 수 있는 범용 프로세서(generic-purpose processor)(예: CPU 또는 application processor)를 의미할 수 있다.

[50] 본 문서에서 사용된 용어들은 단지 특정한 실시예를 설명하기 위해 사용된 것으로, 다른 실시예의 범위를 한정하려는 의도가 아닐 수 있다. 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함할 수 있다.

기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 용어들은 본 문서에 기재된 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가질 수 있다. 본 문서에 사용된 용어들 중 일반적인 사전에 정의된 용어들은, 관련 기술의 문맥상 가지는 의미와 동일 또는 유사한 의미로 해석될 수 있으며, 본 문서에서 명백하게 정의되지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다. 경우에 따라서, 본 문서에서 정의된 용어일지라도 본 문서의 실시예들을 배제하도록 해석될 수 없다.

[51] 이하에서는 도면을 참조하여 본 개시에 대해 더욱 상세히 설명하도록 한다. 다만, 본 개시를 설명함에 있어서, 관련된 공지 기능 혹은 구성에 대한 구체적인 설명이 본 개시의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우 그에 대한 상세한 설명은 생략한다. 도면의 설명과 관련하여, 유사한 구성요소에 대해서는 유사한 참조 부호가 사용될 수 있다.

[52] 본 개시의 실시 예들을 서버리스 컴퓨팅 시스템과 관련된다. 서버리스 컴퓨팅 시스템은 클라이언트로부터 요청된 이벤트에 대한 코드를 실행하여 응답하는 클라우드 컴퓨팅 방식을 제공할 수 있다. 여기서 서버리스, 즉, 서버가 없다는 표현은 해당 이벤트에 대해 별도로 할당된 서버가 없어, 개발자가 인프라 및 플랫폼 관리를 할 필요가 없다는 의미이다.

[53] 도 2는 본 개시의 전자 장치를 포함하는 서버리스 컴퓨팅 시스템을 설명하기 위한 도면이다.

[54] 도 2에 도시된 바와 같이, 서버리스 컴퓨팅 시스템(1000)은 전자 장치(100), 인공지능 서비스 클라이언트(200), 인공지능 모델 클라이언트(300) 및 클라이언트(400)를 포함할 수 있다.

[55] 본 개시의 전자 장치(100)는 클라이언트(400)에게 인공지능 서비스를 제공할 수 있다.

[56] 본 개시에서 인공지능 서비스는 전자 장치(100)를 통하여 클라이언트(400)에게 제공되는 서비스로, 구체적으로, 전자 장치(100)에 포함된 인공지능 모델을 이용하여 클라이언트(400)에게 제공되는 모든 서비스를 지칭하는 것일 수 있다.

[57] 예를 들어, 본 개시의 인공지능 서비스는, 자연어 처리, 기계 번역, 대화시스템, 질의 응답, 음성 인식/합성, 오브젝트 인식, 오브젝트 추적, 영상 검색, 사람 인식, 장면 이해, 공간 이해, 영상 개선, 지식/확률 기반 추론, 최적화 예측, 선호 기반 계획, 추천, 지식 구축(데이터 생성/분류), 지식 관리(데이터 활용), 지식 구축(데이터 생성/분류), 지식 관리(데이터 활용) 등과 같은 기능을 수행하는 인공지능 모델을 이용하여 클라이언트(400)에게 적절한 서비스를 제공하는 서비스 중 어느 하나가 될 수 있다.

[58] 인공지능 서비스를 제공하기 위하여, 전자 장치(100)는 인공지능 서비스 클라이언트(300)로부터 인공지능 서비스 함수(AI service function)를 제공받아 전자 장치(100)에 저장할 수 있다.

[59] 여기에서, 인공지능 서비스 클라이언트(300)는 인공지능 서비스 개발자의 전자

장치에 대응될 수 있다.

[60] 인공지능 서비스 클라이언트(300)는 인공지능 서비스에 대응되는 함수를 생성하고 이를 전자 장치(100)에 제공할 수 있다. 인공지능 서비스에 대응되는 함수는 인공지능 서비스의 특정 작업을 수행하는 코드들의 집합으로, 인공지능 모델 클라이언트(200)가 제공한 model prediction code를 포함할 수 있다.

[61] 구체적으로, 인공지능 서비스 클라이언트(300)는 모델 스토어(model store)(미도시)에 등록된 학습된 인공지능 모델 및 model prediction code를 선택할 수 있다.

[62] 인공지능 서비스 클라이언트(300)는 선택된 인공지능 모델 및 model prediction code를 기초로 특정 작업을 수행하는 인공지능 서비스 함수를 생성할 수 있으며, 이를 전자 장치(100)에 제공할 수 있다.

[63] 한편, 인공지능 모델 클라이언트(200)는 학습된 인공지능 모델 파일, model prediction code 및 인공지능 모델에 대한 operation 정보를 모델 스토어(미도시)에 등록할 수 있다. 학습된 인공지능 모델 파일은 학습된 인공지능 모델의 히든 레이어(hidden layer) 및 인공 지능 모델이 결과 값으로 출력하는 레이블링(labeling) 정보를 포함한 파일이고, model prediction code는 학습된 모델의 입력 값에 대한 결과 값을 얻기 위해 필요한 코드의 집합(또는 프로그램)을 나타내며, 인공지능 모델에 대한 operation 정보는 인공지능 모델을 이용하는데 최소한으로 필요한 CPU, GPU, 메모리 등의 리소스(resource) 정보를 포함할 수 있다.

[64] 또한, 인공지능 모델 클라이언트(200)는 전자 장치(100)가 제공하는 인터페이스(Interface)를 이용하여 학습된 인공지능 모델, model prediction code 및 인공지능 모델에 대한 operation 정보를 모델 스토어(미도시)에 등록할 수 있다.

[65] 인공지능 모델 클라이언트(200)는 인공지능 모델 개발자의 전자 장치에 대응될 수 있다.

[66] 한편, 클라이언트(400)는 인공지능 서비스를 이용하고자 하는 사용자의 전자 장치에 대응될 수 있다.

[67] 클라이언트(400)는 전자 장치(100)에 특정 인공지능 서비스를 요청할 수 있다. 이 경우, 전자 장치(100)는 복수의 인공지능 서비스 클라이언트(300)가 제공한 복수의 인공지능 서비스 함수 중 클라이언트(400)가 요청한 인공지능 서비스에 대응되는 인공지능 서비스 함수를 이용하여 클라이언트(400)에 인공지능 서비스를 제공할 수 있다.

[68] 한편, 전자 장치(100)는 클라우드 서버, 인공지능 서버 등과 같은 서버로 구현될 수 있다. 다만 이에 한정되는 것은 아니고 어떠한 전자 장치로도 구현될 수 있다.

[69] 인공지능 모델 클라이언트(200), 인공지능 서비스 클라이언트(300) 및 클라이언트(400)는 하드웨어 클라이언트, 소프트웨어 클라이언트, 애플리케이션 등일 수 있다. 본 개시의 일 실시 예에 따르면, 인공지능 서비스 클라이언트(200), 인공지능 모델 클라이언트(300) 및 클라이언트(400)는 데스크탑, 태블릿 PC,

노트북, 휴대폰, IoT 디바이스, 웨어러블 디바이스(wearable device) 등과 같은 전자 장치로 구현될 수 있다.

[70] 도 3은 본 개시의 일 실시 예에 따른 전자 장치의 구성을 설명하기 위한 블록도이다.

[71] 도 3을 참조하면, 본 개시의 일 실시 예에 따른 전자 장치(100)는 통신부(110), 메모리(120) 및 프로세서(130)를 포함할 수 있다.

[72] 통신부(110)는 프로세서(130)의 제어에 의해 외부 장치와 데이터 또는 신호를 송수신할 수 있다. 여기에서, 외부 장치는 도 2의 인공지능 모델 클라이언트(200), 인공지능 서비스 클라이언트(300) 및 클라이언트(400)가 될 수 있다.

[73] 전자 장치(100)는 통신부(110)를 통하여, 학습된 인공지능 모델 파일, 인공지능 모델의 model prediction code, 인공지능 모델에 대한 operation 정보, 인공지능 서비스 함수 및 인공지능 서비스에 대한 요청을 수신할 수 있으며, 인공지능 서비스 요청에 대한 응답을 전송할 수 있다.

[74] 통신부(110)는 근거리 통신망(Local Area Network; LAN), 광역 통신망(Wide Area Network; WAN), 부가가치 통신망(Value Added Network; VAN), 이동통신망(mobile radio communication network), 위성 통신망 및 이들의 상호 조합을 통하여 통신을 하게 하는 하나 이상의 구성요소를 포함할 수 있다. 또한, 통신부(110)는 외부 장치 또는 외부 서버와 직접 무선랜(예를 들어, 와이-파이(Wi-Fi)) 등을 이용하여 무선으로 데이터 또는 신호를 송수신할 수 있다.

[75] 메모리(120)는 전자 장치(100)를 구동하고 제어하기 위한 다양한 데이터, 프로그램 또는 어플리케이션을 저장할 수 있다. 메모리(120)에 저장되는 프로그램은 하나 이상의 명령어들을 포함할 수 있다. 메모리(120)에 저장된 프로그램(하나 이상의 명령어들) 또는 어플리케이션은 프로세서(120)에 의해 실행될 수 있다.

[76] 메모리(120)는 예를 들면, 내장 메모리 또는 외장 메모리를 포함할 수 있다. 내장 메모리는, 예를 들면, 휘발성 메모리(예: DRAM(dynamic RAM), SRAM(static RAM), 또는 SDRAM(synchronous dynamic RAM) 등), 비휘발성 메모리(non-volatile Memory)(예: OTPROM(one time programmable ROM), PROM(programmable ROM), EPROM(erasable and programmable ROM), EEPROM(electrically erasable and programmable ROM), mask ROM, flash ROM, 플래시 메모리(예: NAND flash 또는 NOR flash 등), 하드 드라이브, 또는 솔리드 스테이트 드라이브(solid state drive(SSD)) 중 적어도 하나를 포함할 수 있다.

[77] 외장 메모리는 플래시 드라이브(flash drive), 예를 들면, CF(compact flash), SD(secure digital), Micro-SD(micro secure digital), Mini-SD(mini secure digital), xD(extreme digital), MMC(multi-media card) 또는 메모리 스틱(memory stick) 등을 포함할 수 있다. 외장 메모리는 다양한 인터페이스를 통하여 전자 장치(100)와 기능적으로 및/또는 물리적으로 연결될 수 있다.

[78] 메모리(120)는 프로세서(130)에 의해 액세스되며, 프로세서(130)에 의한 데이터의 독취/기록/수정/삭제/갱신 등이 수행될 수 있다. 본 개시에서 메모리라는 용어는 메모리(120), 프로세서(130) 내 롬(미도시), 램(미도시) 또는 전자 장치(100)에 장착되는 메모리 카드(예를 들어, micro SD 카드, 메모리 스틱)(미도시)를 포함할 수 있다.

[79] 본 개시의 일 실시 예에 따라, 메모리(120)는 복수의 함수를 포함하는 데이터베이스를 포함할 수 있다. 여기에서, 데이터베이스는 인공지능 서비스 클라이언트(300)로부터 수신된 함수를 포함할 수 있다.

[80] 프로세서(130)는 전자 장치(100)의 전반적인 동작을 제어하기 위한 구성이다. 예를 들면, 프로세서(130)는 운영 체제(Operating System. OS), 어플리케이션을 구동하여 프로세서(130)에 연결된 다수의 하드웨어 또는 소프트웨어 구성요소들을 제어할 수 있고, 각종 데이터 처리 및 연산을 수행할 수 있다.

[81] 프로세서(130)는 CPU(central processing unit) 또는 GPU(graphics-processing unit)이거나 둘 다일 수 있다. 프로세서(710)는 적어도 하나의 범용 프로세서(general processor), 디지털 신호 프로세서(digital signal processor), ASIC(Application specific integrated circuit), SoC(system on chip), MICOM(Microcomputer) 등으로 구현될 수 있다.

[82] 프로세서(130)는 모델 스토어에 등록된 복수의 인공지능 모델의 속성 및 인공지능 모델을 위한 라이브러리가 로딩된 복수의 컨테이너의 속성에 기초하여, 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다.

[83] 컨테이너에 로딩된 라이브러리에는 컨테이너에 로딩될 인공지능 모델을 학습시키는데 사용된 소프트웨어(가령, 구글사의 TensorFlow ™)의 라이브러리가 포함될 수 있다. 컨테이너는, 학습된 인공지능 모델을 로딩하고 이를 이용하여 컨테이너에 입력된 함수를 실행하기 위하여, 컨테이너에 로딩될 인공지능 모델을 위한 라이브러리를 포함할 수 있다.

[84] 프로세서(130)는 복수의 컨테이너에 할당된 리소스 및 복수의 인공지능 모델에 요구되는 리소스에 기초하여 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다. 복수의 컨테이너는 GPU(Graphic Processing Unit) 또는 CPU(Central Processing Unit)를 기반으로 상기 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수를 실행할 수 있다. GPU 또는 CPU를 기반으로 인공지능 서비스에 대응되는 함수를 실행한다는 것은, GPU 또는 CPU의 리소스를 이용하여 인공지능 서비스에 대응되는 함수를 실행한다는 것을 의미한다.

[85] 프로세서(130)는 컨테이너가 생성될 때, 컨테이너가 CPU를 사용하여 요청된 함수를 실행하는 CPU 컨테이너인지, GPU를 사용하여 요청된 함수를 실행하는 GPU 컨테이너인지 여부 및 컨테이너에 바인딩(binding)된 CPU/GPU 메모리 등과 같이 컨테이너에 할당된 리소스 정보 등을 획득할 수 있다.

[86] 그리고, 프로세서(130)는 모델 스토어(미도시)에 등록된 인공지능 모델에 대한 operation 정보를 기초로, 인공지능 모델에 요구되는 전자 장치(100)의

리소스(가령, CPU, GPU, 메모리 등) 정보를 획득할 수 있다.

[87] 프로세서(130)는 복수의 컨테이너 각각에 할당된 리소스 및 각 컨테이너에 로딩된 인공지능 모델에 요구되는 리소스에 기초하여 각 컨테이너에서 사용 가능한 리소스를 판단하고, 판단된 리소스에 기초하여 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다.

[88] 가령, 복수의 컨테이너 중 제1 컨테이너에 1G Bytes의 GPU 메모리가 바인딩(binding)되어 있고, 제1 컨테이너에 로딩되어 있는 제1 인공지능 모델이 300M Bytes의 GPU 메모리를 필요로 하는 경우, 프로세서(130)는 제1 컨테이너에서 사용 가능한 GPU 메모리가 700M Bytes라고 판단하고, 이에 기초하여 제2 인공지능 모델이 제1 컨테이너에 로딩 가능한지 여부를 판단할 수 있다. 가령, 제2 인공지능 모델이 필요로 하는 리소스가 300M Bytes 인 경우, 프로세서(130)는 제2 인공지능 모델이 제1 컨테이너에 로딩될 수 있다고 판단할 수 있다. 그러나 제2 인공지능 모델이 필요로 하는 리소스가 800M Bytes 인 경우, 프로세서(130)는 제2 인공지능 모델이 제1 컨테이너에 로딩될 수 없다고 판단할 수 있다.

[89] 프로세서(130)는 인공지능 모델 클라이언트(200)에 의하여 학습된 인공지능 모델이 모델 스토어(미도시)에 등록되는 경우, 복수의 컨테이너의 속성 및 모델 스토어(미도시)에 등록된 인공지능 모델의 속성에 기초하여 등록된 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다.

[90] 이때, 프로세서(130)는 인공지능 모델 클라이언트(200)에 인터페이스(interface)를 제공하여 인공지능 모델 클라이언트(200)가 제공된 인터페이스에 따라 학습된 인공지능 모델 파일 및 인공지능 모델의 model prediction code를 등록하도록 할 수 있다.

[91] 학습된 인공지능 모델 파일은 모델 바이너리 파일(model binary file) 및 해당 모델 파일을 실행하기 위한 코드(code)가 필요한데, 이는 인공지능 모델을 학습시키는 소프트웨어에 따라 다양할 수 있다.

[92] 이에 따라, 프로세서(130)는 인공지능 모델 클라이언트(200)가 동일한 인터페이스를 통하여 인공지능 모델 파일 및 model prediction code를 모델 스토어(미도시)에 등록하도록 하여 복수의 인공지능 모델 클라이언트(200)에 의해 제공되는 다양한 코드(code)가 컨테이너 상에 포함된 동일한 라이브러리를 바탕으로 실행될 수 있도록 할 수 있다.

[93] 프로세서(130)는 인터페이스를 이용하여 모델에 대한 함수 요청이 있을 때 실행되는 코드와 함수 요청 전에 인공지능 모델을 컨테이너에 로딩 하는 코드를 구분하여 실행할 수 있다. 즉, 프로세서(130)는 인터페이스를 이용하여 컨테이너가 인공지능 서비스 요청에 대한 함수를 획득하기 이전에 인공지능 모델을 컨테이너에 로딩할 수 있다. 컨테이너가 미리 인공지능 모델을 로딩하면 함수를 전달 받은 컨테이너가 인공지능 모델을 로딩하는 단계를 생략할 수 있다는 점에서, 컨테이너가 함수를 실행하기 시간을 절약할 수 있다.

[94] 프로세서(130)는 컨테이너에 로딩된 라이브러리를 바탕으로 컨테이너에 인공지능 모델을 로딩할 수 있다. 상술한 바와 같이, 컨테이너에 로딩된 라이브러리는 인공지능 모델을 학습시키는데 사용된 소프트웨어의 라이브러리라는 점에서, 즉, 인공지능 모델을 학습시킨 라이브러리와 컨테이너에 포함된 라이브러리가 동일하다는 점에서, 프로세서(130)는 컨테이너에 학습된 인공지능 모델을 로딩할 수 있다.

[95] 한편, 프로세서(130)는 통신부(110)를 통하여 클라이언트(400)로부터 인공지능 서비스에 대한 요청이 수신되면, 메모리(120)에 저장된 데이터베이스로부터 요청된 인공지능 서비스에 대응되는 함수를 획득할 수 있다.

[96] 인공지능 서비스에 대응되는 함수는 인공지능 서비스 클라이언트(300)에 의해 생성된 것으로, 인공지능 서비스 클라이언트(300)는 모델 스토어에 등록된 학습된 인공지능 모델을 선택하고 선택된 인공지능 모델의 model prediction code를 포함하는 인공지능 서비스 함수를 생성한다는 점에서, 인공지능 서비스에 대응되는 함수에는 인공지능 모델에 관한 정보가 포함될 수 있다.

[97] 프로세서(130)는 획득된 함수에 포함된 인공지능 모델의 정보를 기초로 복수의 컨테이너 중에서 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단할 수 있다.

[98] 구체적으로, 프로세서(130)는 획득된 함수에 포함된 인공지능 모델의 model prediction code를 기초로 인공지능 서비스에 대응되는 인공지능 모델을 판단하고, 인공지능 모델이 로딩된 컨테이너 정보를 포함하는 모델 리스트를 기초로 판단된 인공지능 모델이 로딩된 컨테이너를 판단할 수 있다. 여기에서, 모델 리스트는 복수의 컨테이너 중 모델 스토어에 등록된 적어도 하나의 학습된 인공지능 모델이 로딩된 컨테이너를 식별하기 위한 것으로, 학습된 인공지능 모델이 컨테이너에 로딩될 때 모델 리스트가 갱신될 수 있다.

[99] 예를 들어, 프로세서(130)가 통신부(110)를 통하여 클라이언트(400)로부터 얼굴인식 서비스에 대한 요청을 수신하면, 프로세서(130)는 데이터베이스로부터 얼굴인식 서비스에 대응되는 얼굴 인식 함수를 획득할 수 있으며, 획득된 얼굴인식 함수에 포함된 인공지능 모델의 model prediction code를 기초로 얼굴인식 서비스에 대응되는 얼굴인식 인공지능 모델을 판단하고, 모델 리스트를 이용하여 판단된 얼굴인식 인공지능 모델이 로딩된 컨테이너를 판단할 수 있다.

[100] 프로세서(130)는 획득된 함수에 실행하기 위한 인공지능 모델이 로딩되어 있다고 판단된 컨테이너에서 획득된 함수를 실행할 수 있다.

[101] 구체적으로, 프로세서(130)는 인공지능 서비스를 실행하기 위한 인공지능 모델이 로딩된 컨테이너에서 컨테이너에 로딩된 라이브러리를 바탕으로 획득된 함수를 실행할 수 있다. 상술한 바와 같이, 컨테이너에 로딩된 라이브러리는 컨테이너에 로딩된 인공지능 모델을 학습시키는데 사용된 소프트웨어의 라이브러리라는 점에서, 컨테이너는 로딩된 라이브러리를 이용하여 학습된

인공지능 모델에 입력된 입력 값으로부터 결과 값(추론 값)을 획득할 수 있다.

[102] 즉, 프로세서(130)는 컨테이너에 로딩된 인공지능 모델로부터 클라이언트(400)의 요청에 대한 데이터를 획득할 수 있다.

[103] 그리고, 프로세서(130)는 획득된 데이터를 통신부(110)를 통해 클라이언트(400)로 전송할 수 있다.

[104] 한편, 프로세서(130)는 각 컨테이너에서 클라이언트 장치(400)에서 요청된 인공지능 서비스에 대응되는 함수의 실행 시간에 대한 정보를 수집할 수 있다.

[105] 구체적으로, 프로세서(130)는 함수를 실행한 컨테이너로부터 함수를 실행하는데 소요된 시간에 관한 정보를 수신할 수 있다.

[106] 프로세서(130)는 수신된 정보를 바탕으로 GPU 기반의 컨테이너 및 CPU 기반의 컨테이너 중에서 함수가 실행될 컨테이너를 판단할 수 있다. 구체적으로, 프로세서(130)는 추후 실행된 함수와 동일한 함수를 실행할 경우, 해당 함수가 PU 기반의 컨테이너에서 실행되는 것이 효율적인지 또는 CPU 기반의 컨테이너에서 실행되는 것이 효율적인지 여부를 판단할 수 있다.

[107] 가령, 프로세서(130)는 GPU 기반의 컨테이너에서 특정 함수의 수행 시간이 기 설정된 값 미만이면, 해당 함수는 추후 CPU 기반의 컨테이너에서 실행되는 것이 적절하다고 판단할 수 있다. 그리고, 추후 동일한 인공지능 서비스에 대응되는 함수를 수신하는 경우, 해당 함수를 GPU 컨테이너가 아닌 CPU 컨테이너에서 실행할 수 있다.

[108] 다만, 이는 일 실시 예이며, 프로세서(130)는 CPU 기반의 컨테이너에서 특정 함수의 수행 시간이 기 설정된 값 이상이면, 해당 함수는 추후 GPU 기반의 컨테이너에서 실행되는 것이 적절하다고 판단하여, 추후 인공지능 서비스에 대응되는 함수를 수신하는 경우 해당 함수를 GPU 컨테이너에서 실행할 수 있다.

[109] 이와 같이, 하나의 컨테이너는 동일한 인공지능 서비스에 대응되는 함수를 여러 번 수신할 수 있다. 이 경우, 프로세서(130)는 동일한 컨테이너가 동일한 인공지능 서비스에 대한 함수를 수행하여 인공지능 서비스 요청에 대한 응답을 제공하도록 할 수 있다. 즉, 프로세서(130)는 동일한 인공지능 서비스에 대한 요청이 오는 경우, 하나의 컨테이너에 바인딩된 리소스를 재사용 함으로써, 동일한 요청에 대한 응답 시간을 단축시킬 수 있다.

[110] 한편, 프로세서(130)는 기 설정된 시간 동안 하나의 컨테이너에 요청되는 인공지능 서비스에 대한 횟수가 기 설정된 횟수 이상인 경우, 동일한 인공지능 서비스를 수행하는 컨테이너를 추가로 생성할 수 있다. 즉, 프로세서(130)는 기 설정된 시간 동안 하나의 컨테이너가 동일한 인공지능 서비스에 대응되는 함수를 기 설정된 횟수 이상 수행하는 경우, 해당 함수를 실행하는 컨테이너를 여러 개 생성할 수 있다.

[111] 한편, 프로세서(130)는 컨테이너가 제1 상태인 상태에서 획득된 함수를 실행하면 컨테이너를 제2 상태로 판단할 수 있다. 여기에서, 제1 상태는 컨테이너에 인공지능 모델이 로딩되지 않은 상태 또는 컨테이너에 적어도

하나의 학습된 인공지능이 로딩되고, 컨테이너가 로딩된 적어도 하나의 학습된 인공지능 모델을 이용하여 함수를 실행하지 않은 상태로 프리웜(pre-warm)상태를 나타내는 것일 수 있다. 그리고, 제2 상태는 컨테이너가 로딩된 적어도 하나의 학습된 인공지능 모델을 이용하여 함수를 실행한 상태로 웜(warm) 상태를 의미할 수 있다.

[112] 즉, 프로세서(130)는 프리웜 상태인 컨테이너가 함수를 실행하면 컨테이너가 웜 상태로 전환된 것으로 판단할 수 있다.

[113] 그리고, 프로세서(130)는 복수의 컨테이너 중 제1 상태인 컨테이너의 개수가 기 설정된 수 미만이면 새로운 컨테이너를 생성할 수 있다. 즉, 프로세서(130)는 제1 상태인 컨테이너의 개수를 기설정된 개수로 유지할 수 있다. 따라서, 제1 상태인 컨테이너가 함수를 실행하여 제2 상태로 되면, 프로세서(130)는 제1 상태인 새로운 컨테이너를 생성할 수 있다.

[114] 예를 들어, 전자 장치(100)가 제1 상태(프리웜 상태)인 컨테이너를 3개 유지한다고 가정하자. 프로세서(130)는 3개의 컨테이너(제1, 제2 및 제3 컨테이너) 중 제1 인공지능 모델이 로딩된 제1 컨테이너에서 요청된 서비스에 대응되는 함수가 제1 인공지능 모델을 기초로 실행되어 제1 컨테이너가 제1 상태에서 제2 상태가 된 것으로 판단되면, 제1 상태인 컨테이너는 2개(제2 및 제3 컨테이너)인 것으로 판단할 수 있다. 이 경우, 프로세서(130)는 제1 상태인 새로운 컨테이너(제4 컨테이너)를 생성할 수 있다.

[115] 한편, 프로세서(130)는 획득된 함수를 실행한 컨테이너(즉, 제2 상태인 컨테이너)가 기 설정된 시간 동안 컨테이너에 포함된 모델을 이용하는 인공지능 서비스에 대한 요청을 수신하지 않은 경우, 획득된 함수를 실행한 컨테이너를 킬(Kill) 할 수 있다.

[116] 즉, 프로세서(130)는 함수를 실행하여 제2 상태가 된 컨테이너가 기 설정된 시간 동안 컨테이너에 포함된 적어도 하나의 모델을 이용하는 함수를 획득하지 않은 경우, 제2 상태가 된 컨테이너를 킬 할 수 있다.

[117] 이 경우, 프로세서(130)는 킬 되는 컨테이너에 포함된 적어도 하나의 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다.

[118] 상술한 바와 같이, 전자 장치(100)에는 제1 상태인 컨테이너가 기 설정된 수만큼 유지되어야 한다는 점에서, 인공지능 모델이 로딩된 컨테이너가 삭제되더라도 적어도 기 설정된 수만큼의 컨테이너가 존재할 수 있다.

[119] 프로세서(130)는 킬 된 컨테이너를 제외한 나머지 컨테이너의 속성 및 킬된 컨테이너에 로딩되어 있던 인공지능 모델의 속성에 기초하여 킬된 컨테이너에 로딩되어 있던 인공지능 모델이 새롭게 로딩될 컨테이너를 판단할 수 있다.

[120] 구체적으로, 프로세서(130)는 킬 된 컨테이너를 제외한 나머지 컨테이너에서 사용 가능한 리소스 및 킬 된 컨테이너에 로딩되어 있던 인공지능 모델에 요구되는 리소스에 기초하여, 킬 된 컨테이너에 로딩되어 있던 인공지능 모델이 새롭게 로딩될 컨테이너를 판단할 수 있다.

[121] 예를 들어, 전자 장치(100)에 제1 내지 제5 컨테이너가 존재하고, 제1 컨테이너에는 제1 및 제2 인공지능 모델이, 제2 컨테이너에는 제3 인공지능 모델 및 제4 인공지능 모델이, 제3 컨테이너에는 제5 인공지능 모델이 존재하며, 제4 컨테이너 및 제5 컨테이너에는 인공지능 모델이 로딩되어 있지 않다고 가정하자. 그리고, 제1 컨테이너 및 제2 컨테이너는 함수를 실행하여 제2 상태이며, 이 중 제1 컨테이너가 기 설정된 시간(가령, 1분)동안 인공지능 서비스에 대응되는 함수를 수신하지 않았다면, 프로세서(130)는 제1 컨테이너를 킬 할 수 있다.

[122] 이때, 제1 컨테이너에 포함된 제1 및 제2 인공지능 모델에 필요한 리소스가 각각 600M Bytes 및 300M Bytes 이고, 제2 컨테이너가 사용 가능한 리소스가 0 M Bytes, 제3 컨테이너가 사용 가능한 리소스가 400M Bytes, 제 4 및 제5 컨테이너가 사용 가능한 리소스가 각각 1G Bytes인 경우, 프로세서(130)는 제1 인공지능 모델을 제4 컨테이너에 로딩하고, 제2 인공지능 모델을 제3 컨테이너에 할당할 수 있다. 다만, 이는 일 실시 예이며, 프로세서(130)가 제1 인공지능 모델 및 제2 인공지능 모델을 제4 컨테이너에 로딩하거나, 제1 인공지능 모델을 제4 컨테이너에 로딩하고, 제2 인공지능 모델을 제5 컨테이너에 로딩할 수 있음은 물론이다.

[123] 한편, 본 개시의 다양한 실시 예에 따라, 프로세서(130)는 도 4 및 도 5에서 후술할 모델 코디네이터(Model Coordinator)(510), 인보커(invoker)(520), GPU 런타임 컨테이너(GPU Runtime Container)(530), CPU 런타임 컨테이너(CPU Runtime Container)(540), 컨트롤러(controller)(550), 큐(Queue)(560), 모델 스토어(570), 강화 훈련 모듈(590) 중 적어도 하나의 기능을 수행할 수 있다.

[124] 도 4 및 도 5는 본 개시에 따른 전자 장치를 설명하기 위한 도면이다. 구체적으로, 도 4는 본 개시의 일 실시 예에 따른 컨테이너를 설명하기 위한 도면이다.

[125] 인보커(520)는 컨테이너(530, 540)를 생성할 수 있다. 이때, 생성되는 컨테이너는 CPU 리소스를 사용하는 CPU 기반의 컨테이너(540)이거나 GPU 리소스를 사용하는 GPU 기반의 컨테이너(530)가 될 수 있다.

[126] 인보커(520)에 의해 생성된 컨테이너(530, 540)은 모델 코디네이터(510)에 컨테이너 정보를 제공할 수 있다. 여기에서, 컨테이너 정보는 컨테이너에 할당된 리소스 정보, 컨테이너가 CPU/GPU 컨테이너인지 여부 등을 포함할 수 있다.

[127] 인보커(520)에 의해 생성된 컨테이너에는 인공지능 모델을 위한 라이브러리가 포함될 수 있다. 인공지능 모델을 위한 라이브러리를 바탕으로, 컨테이너(530)는 학습된 인공지능 모델을 로딩할 수 있으며, 로딩된 인공지능 모델을 이용하여 함수를 실행할 수 있다.

[128] 인공지능 모델이 로딩되지 않은 컨테이너 또는 적어도 하나의 인공지능 모델이 로딩되어 있으나 로딩된 인공지능 모델을 이용하여 함수를 실행하지 않은 컨테이너는 프리 웜(pre-warm) 컨테이너로 판단될 수 있다. 반면, 로딩된

인공지능 모델을 이용하여 함수를 실행한 컨테이너는 웜(warm) 컨테이너로 판단될 수 있다.

[129] 본 개시의 일 실시 예에 따른 전자 장치(100)는 프리 웜 컨테이너를 기 설정된 개수만큼 유지할 수 있다. 컨테이너는 인공지능 서비스에 대응되는 함수를 수신하기 전의 프리 웜 컨테이너 상태에서 학습된 인공지능 모델을 로딩할 수 있다.

[130] 그리고, 프리 웜 컨테이너가 인공지능 서비스에 대응되는 함수를 수신하면 함수를 실행하고 웜 컨테이너로 전환될 수 있다. 이 경우, 컨테이너가 인공지능 서비스에 대응되는 인공지능 모델을 미리 로딩한 상태라는 점에서, 컨테이너가 함수를 실행하는 시간은 절약될 수 있다.

[131] 또한, 전자 장치(100)가 기 설정된 개수의 프리 웜 컨테이너를 유지하여야 한다는 점에서, 프리 웜 컨테이너가 웜 컨테이너로 전환되면 인보커(520)는 새로운 컨테이너를 생성하고, 생성된 컨테이너는 모델 코디네이터(510)에 컨테이너 정보를 제공할 수 있다.

[132] 한편, 하나의 컨테이너에는 복수의 인공지능 모델이 로딩될 수 있다.

[133] 하나의 컨테이너에 포함된 복수의 인공지능 모델에 대응되는 함수가 기 설정된 시간 동안 실행되지 않는 경우, 인보커(520)는 해당 컨테이너를 킬(Kill)할 수 있다.

[134] 인보커(520)에 의해 컨테이너가 킬 되는 경우, 모델 코디네이터(510)는 킬 된 컨테이너에 포함되어 있던 적어도 하나의 인공지능 모델이 다른 컨테이너에 로딩되도록 할 수 있다.

[135] 구체적으로, 모델 코디네이터(510)는 킬된 컨테이너에 포함되어 있던 적어도 하나의 인공지능 모델이 필요로 하는 리소스 정보 및 킬된 컨테이너를 제외한 나머지 컨테이너에서 사용가능한 리소스 정보를 기초로, 킬 된 컨테이너에 포함되어 있던 적어도 하나의 인공지능 모델이 로딩될 다른 컨테이너를 식별할 수 있다. 그리고, 모델 코디네이터(510)는 식별된 컨테이너에 킬된 컨테이너에 포함되어 있던 적어도 하나의 인공지능 모델이 로딩 되도록, 킬된 컨테이너에 포함되어 있던 인공지능 모델의 정보를 식별된 컨테이너에 전달할 수 있다.

[136] 그리고, 모델 코디네이터(510)로부터 킬된 컨테이너에 포함되어 있던 인공지능 모델의 정보를 수신한 컨테이너는 수신된 인공지능 모델의 정보에 대응되는 인공지능 모델을 로딩할 수 있다.

[137] 전자 장치(100)는 상술한 컨테이너를 기반으로 인공지능 서비스를 제공할 수 있다.

[138] 도 5는 본 개시의 일 실시 예에 따라 클라이언트의 요청에 대응되는 인공지능 서비스를 제공하는 전자 장치를 설명하기 위한 도면이다.

[139] 모델 스토어(210)는 복수의 학습된 인공지능 모델을 포함한다. 그리고, 모델 스토어(570)는 인공지능 모델 클라이언트(200)에 의해 등록된 새로운 인공지능 모델 또는 기존에 모델 스토어(210)에 등록된 인공지능 모델을 관리할 수 있다.

구체적으로, 모델 스토어(210)는 새로운 인공지능 모델이 등록되는 경우 또는 기존에 등록된 인공지능 모델이 업데이트되는 경우 등록 정보 또는 업데이트 정보를 모델 코디네이터(510)에 제공할 수 있다.

[140] 모델 코디네이터(510)는 인공지능 모델 클라이언트(200)에 의하여 모델 스토어(570)에 학습된 인공지능 모델이 등록되면, 컨테이너(530, 540)가 모델 스토어(570)에 등록된 학습된 인공지능 모델을 로딩하도록 할 수 있다.

[141] 모델 코디네이터(510)는 모델 스토어(570)에 학습된 인공지능 모델이 등록되면, 모델 스토어(570)에 등록된 인공지능 모델 정보, 복수의 컨테이너에 현재 로딩된 모델 정보 및 현재 요청된 인공지능 서비스에 대응되는 함수를 처리하고 있는 컨테이너의 상태 정보 등을 기초로, 학습된 인공지능 모델이 로딩될 컨테이너(530, 540)를 선택할 수 있다.

[142] 예를 들어, 모델 코디네이터(510)는 모델 스토어(570)에 등록된 인공지능 모델의 operation 정보(가령, 인공지능 모델에 요구되는 리소스 정보), 복수의 컨테이너 각각에 할당된 리소스 정보, 복수의 컨테이너에 현재 로딩된 모델 정보를 기초로 각 컨테이너에서 사용 가능한 리소스를 판단하고, 이를 기초로 모델 스토어(570)에 등록된 인공지능 모델이 로딩될 컨테이너를 선택할 수 있다.

[143] 그리고, 모델 코디네이터(510)는 판단된 컨테이너에게 인공지능 모델의 정보를 제공할 수 있다.

[144] 이 경우, 컨테이너(530, 540)는 모델 코디네이터(510)로부터 제공된 정보에 기초하여 모델 스토어(570)로부터 로딩할 인공지능 모델을 판단하고, 해당 인공지능 모델을 모델 스토어(570)로부터 로딩할 수 있다. 이때, 컨테이너(530, 540)는 컨테이너에 하나의 인공지능 모델이 로딩된 상태라도, 모델 코디네이터(510)로부터 추가적으로 로딩해야 하는 모델들의 리스트 정보를 제공받을 수 있다. 여기에서 모델들의 리스트 정보는 모델 코디네이터(510)가 모델 스토어(210)로부터 제공 받은 모델 스토어(210)에 저장되어 있는 학습된 모델들의 리스트 정보를 포함할 수 있다.

[145] 구체적으로, 모델 코디네이터(510)로부터 로딩될 인공지능 모델의 정보를 수신한 컨테이너(530, 540)는 컨테이너에 포함된 다이나믹 로더(Dynamic Loader)를 이용하여 컨테이너(530, 540)에 인공지능 모델을 로딩할 수 있다.

[146] 그리고, 모델 코디네이터(510)는 인공지능 모델의 정보 및 인공지능 모델이 로딩된 컨테이너 정보를 인보커(520)에게 제공할 수 있다. 이 경우, 후술하는 바와 같이, 인보커(520)는 모델 코디네이터(510)로부터 제공받은 정보를 기초로, 요청된 인공지능 서비스에 대응되는 함수를 인공지능 서비스에 대응되는 인공지능 모델이 포함된 컨테이너에 제공할 수 있다.

[147] 한편, 컨트롤러(550)는 클라이언트(400)로부터 인공지능 서비스에 대한 요청을 수신할 수 있다. 이때, 컨트롤러(550)는 클라이언트(400)가 인공지능 서비스를 제공받기 한 인공지능 모델에 대한 정보와 함께 인공지능 서비스의 입력 데이터를 클라이언트(400)로부터 수신할 수 있다. 가령, 클라이언트(400)가

얼굴인식 서비스를 요청한 경우, 컨트롤러(550)는 클라이언트(400)로부터 얼굴인식 서비스에 대한 요청과 함께 얼굴인식 서비스에 대한 입력 데이터로 이미지 파일을 수신할 수 있다.

[148] 이 경우, 컨트롤러(550)는 클라이언트(400)로부터 수신된 요청 및 입력 데이터를 데이터베이스(580)에 저장할 수 있다. 이때, 데이터베이스(580)에는 등록된 인공지능 모델을 이용하여 제공할 수 있는 모든 작업에 대한 식별 정보가 저장되어 있을 수 있다.

[149] 컨트롤러(550)는 데이터베이스(580)에서 클라이언트(400)로부터 수신된 인공지능 서비스 요청을 수행하기 위해 요구되는 작업의 식별 정보를 획득하여 이를 큐(560)에 전달할 수 있다. 한편, 인보커(520)는 복수 개 존재할 수 있는데, 컨트롤러(550)는 복수의 인보커 중 획득된 함수가 할당될 인보커(520)를 지정할 수 있다.

[150] 구체적으로, 컨트롤러(550)는 인보커(520)에 할당된 작업의 양을 판단하고, 판단된 작업을 전달할 인보커(520)를 식별할 수 있다. 이를 위하여, 인보커(520)는 정기적으로 컨트롤러(550)에게 인보커(520)의 상태 정보를 제공할 수 있다.

[151] 그리고, 컨트롤러(550)는 데이터베이스(580)로부터 획득된 작업의 식별 정보 및 지정된 인보커 정보를 큐(560)에 제공할 수 있다.

[152] 그리고, 큐(560)는 컨트롤러(550)로부터 수신된 작업의 식별 정보를 지정된 인보커(560)에게 제공할 수 있다.

[153] 큐(560)로부터 작업 정보를 수신한 인보커(560)는 데이터베이스(580)를 탐색하여 작업의 식별 정보와 매칭되는 작업을 획득할 수 있다. 여기에서, 작업의 식별 정보와 매칭되는 작업은 클라이언트(400)로부터 요청된 인공지능 서비스를 수행하기 위하여 필요한 작업을 의미한다. 또한, 인보커(560)는 데이터베이스(580)로부터 획득된 작업을 수행하기 위한 코드 및 클라이언트(400)가 저장한 입력 데이터를 획득할 수 있다.

[154] 인보커(560)는 획득된 함수의 코드를 컨테이너(530, 540)에 전달할 수 있다. 이를 위하여, 상술한 바와 같이, 인보커(560)는 획득된 함수를 수행 또는 실행하기 위해 필요한 인공지능 모델이 로딩된 컨테이너에 관한 정보를 모델 코디네이터(510)로부터 수신할 수 있다. 그리고, 인보커(560)는 획득된 함수의 코드 및 클라이언트(400)에 의해 데이터베이스(580)에 저장된 입력 데이터를 해당 인공지능 모델이 로딩된 컨테이너로 전달할 수 있다.

[155] 이 경우, 인보커(560)로부터 함수의 코드를 받은 컨테이너(530, 540)는 컨테이너에 로딩된 라이브러리를 바탕으로, 요청된 인공지능 서비스에 대응되는 함수를 실행할 수 있다. 그리고, 컨테이너(530, 540)는 컨트롤러(550)를 통하여, 함수를 실행한 후 획득한 결과 값을 클라이언트(400)에 제공할 수 있다.

[156] 컨테이너는 적어도 하나의 인공지능 모델을 포함할 수 있다. 상술한 바와 같이, 컨테이너에 로딩된 인공지능 모델은 전자 장치(100)가 제공하는 인터페이스를

통하여 모델 스토어에 등록된 인공지능 모델이다. 또한 인보커(560)가 획득한 함수의 코드는 인공지능 서비스 클라이언트(300)가 제공한 함수 코드로 인공지능 서비스 클라이언트(300)는 모델 스토어에 등록된 인공지능 모델 및 model prediction code를 이용하여 함수를 생성하였다.

[157] 즉, 컨테이너(530, 540)가 인보커(560)로부터 획득한 함수는 컨테이너에 로딩된 인공지능 모델과 동일한 인터페이스를 통하여 작성되었다는 점에서, 컨테이너(530,540)는 컨테이너에 로딩된 인공지능 모델을 이용하여 함수를 실행할 수 있다. 구체적으로, 컨테이너는 클라이언트(400)에 의해 제공된 입력 데이터를 인공지능 모델의 입력 값으로 하여 결과 값을 획득할 수 있다. 여기에서 결과 값은 인공지능 모델이 함수를 실행하여 얻은 결과 값을 나타낸다.

[158] 한편, 컨테이너(530, 540)는 함수의 실행 시간 정보(execution time log)를 모델 코디네이터(510)에 전송할 수 있으며, 모델 코디네이터(510)는 컨테이너의 함수 실행 시간 정보를 강화 훈련 모듈(570)에 제공할 수 있다.

[159] 강화 훈련 모듈(570)은 각각의 컨테이너에서 함수를 실행한 시간에 관한 정보를 수신하고, 추후 동일한 함수가 컨테이너에서 실행될 경우 GPU 기반의 컨테이너(530)에서 실행되어야 할지, CPU 기반의 컨테이너(540)에서 실행되어야 할지를 판단할 수 있다. 예를 들어, GPU 기반의 컨테이너가 함수를 실행한 시간이 기 설정된 시간 미만인 경우, 강화 훈련 모듈(570)은 해당 함수가 다시 호출될 경우, 해당 함수는 CPU 기반의 컨테이너에서 실행되는 것이 효율적이라고 판단할 수 있다. 또는, CPU 기반의 컨테이너가 함수를 실행한 시간이 기 설정된 시간 이상인 경우, 강화 훈련 모듈(570)은 해당 함수가 다시 호출될 경우, 해당 함수는 GPU 기반의 컨테이너에서 실행되는 것이 효율적이라고 판단할 수 있다.

[160] 그리고, 강화 훈련 모듈(570)은, 실행 시간 정보를 판단하여 함수에 대응되는 컨테이너 정보가 변경된 경우, 데이터베이스(580)에 저장된 해당 함수의 정보를 업데이트 할 수 있다.

[161] 이에 따라, 추후 동일한 함수에 대응되는 인공지능 서비스에 대한 요청이 수신되면, 컨트롤러(550)는 데이터베이스로부터 업데이트된 함수의 정보를 수신하고, 해당 함수는 함수 수행 시간이 반영된 컨테이너에서 실행될 수 있다.

[162] 도 6은 본 개시의 일 실시 예에 따른 전자 장치를 제어하는 방법을 설명하기 위한 흐름도이다

[163] 인공지능 모델을 위한 라이브러리가 로딩된 복수의 컨테이너의 속성 및 모델 스토어에 등록된 복수의 인공지능 모델의 속성에 기초하여, 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다(S610).

[164] 구체적으로, 복수의 컨테이너에 할당된 리소스 및 상기 복수의 인공지능 모델에 요구되는 리소스에 기초하여 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다.

[165] 더욱 구체적으로, 복수의 컨테이너 각각에 할당된 리소스 및 각 컨테이너에

로딩된 인공지능 모델에 요구되는 리소스에 기초하여 각 컨테이너에서 사용 가능한 리소스를 판단하고, 판단된 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단할 수 있다.

[166] 모델 스토어에 인공지능 모델이 등록되는 경우, 복수의 컨테이너의 속성 및 등록된 인공지능 모델의 속성에 기초하여 등록된 인공지능 모델이 로딩될 컨테이너를 판단할 수 있다.

[167] 한편, 컨테이너에 로딩된 라이브러리를 바탕으로 컨테이너에 인공지능 모델을 로딩할 수 있다(S620).

[168] 클라이언트로부터 인공지능 서비스에 대한 요청이 수신되면 복수의 함수를 포함하는 데이터베이스로부터 요청된 인공지능 서비스에 대응되는 함수를 획득할 수 있다(S630).

[169] 복수의 컨테이너 중에서 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단할 수 있다(S640).

[170] 복수의 컨테이너는 GPU(Graphic Processing Unit) 또는 CPU(Central Processing Unit)를 기반으로 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수를 실행할 수 있다.

[171] 컨테이너에서 컨테이너에 로딩된 라이브러리를 바탕으로 획득된 함수를 실행하여, 판단된 컨테이너에 로딩된 인공지능 모델로부터 상기 요청에 대한 데이터를 획득할 수 있다(S650).

[172] 한편, 컨테이너가 제1 상태인 상태에서 획득된 함수를 실행하면 컨테이너의 상태가 제2 상태인 것으로 판단할 수 있다.

[173] 그리고, 복수의 컨테이너 중 제1 상태인 컨테이너의 개수가 기 설정된 개수 미만이면, 새로운 컨테이너를 생성할 수 있다.

[174] 반면, 획득된 함수를 실행한 컨테이너가 기 설정된 시간 동안 컨테이너에 포함된 모델을 이용하는 인공지능 서비스에 대한 요청을 수신하지 않은 경우, 획득된 함수를 실행한 컨테이너를 킬 할 수 있다.

[175] 킬 된 컨테이너를 제외한 나머지 컨테이너의 속성 및 킬 된 컨테이너에 로딩되어 있던 인공지능 모델의 속성에 기초하여 킬 된 컨테이너에 로딩되어 있던 인공지능 모델이 새롭게 로딩될 컨테이너를 판단할 수 있다.

[176] 그리고, 획득된 데이터를 클라이언트 장치로 전송할 수 있다(S660).

[177] 한편, 각 컨테이너에서의 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수의 실행 시간에 대한 정보를 수집할 수 있다.

[178] 그리고, 수집된 정보를 바탕으로 GPU 기반의 컨테이너 및 CPU 기반의 컨테이너 중에서 함수가 실행될 컨테이너를 판단할 수 있다.

[179] 상술한 본 개시의 다양한 실시 예들에 따르면, 컨테이너에 인공지능 서비스에 대응되는 함수가 전달되기 이전에 미리 인공지능 서비스에 대응되는 인공지능 모델을 로딩하여, 컨테이너가 함수를 실행하는 시간을 단축시킬 수 있다. 또한, 기 설정된 시간동안 동일한 인공지능 서비스에 대한 요청이 수신되는 경우,

컨테이너에 바인딩된 동일한 리소스를 재사용함으로써, 응답 시간(response time)을 단축시킬 수 있다. 그리고, 하나의 인공지능 서비스에 대하여 기본적으로 하나의 컨테이너만을 제공한다는 점에서 전자 장치의 리소스를 효율적으로 사용할 수 있다.

[180] 이상에서 설명된 다양한 실시 예들은 소프트웨어(software), 하드웨어(hardware) 또는 이들의 조합으로 구현될 수 있다. 하드웨어적인 구현에 의하면, 본 개시에서 설명되는 실시 예들은 ASICs(Application Specific Integrated Circuits), DSPs(digital signal processors), DSPDs(digital signal processing devices), PLDs(programmable logic devices), FPGAs(field programmable gate arrays), 프로세서(processors), 제어기(controllers), 마이크로 컨트롤러(micro-controllers), 마이크로 프로세서(microprocessors), 기타 기능 수행을 위한 전기적인 유닛(unit) 중 적어도 하나를 이용하여 구현될 수 있다. 특히, 이상에서 설명된 다양한 실시 예들은 전자 장치(100)의 프로세서(130)에 의해 구현될 수 있다. 소프트웨어적인 구현에 의하면, 본 명세서에서 설명되는 절차 및 기능과 같은 실시 예들은 별도의 소프트웨어 모듈들로 구현될 수 있다. 상기 소프트웨어 모듈들 각각은 본 명세서에서 설명되는 하나 이상의 기능 및 작동을 수행할 수 있다.

[181] 본 개시의 다양한 실시 예들은 기기(machine)(예: 컴퓨터)로 읽을 수 있는 저장 매체(machine-readable storage media)에 저장될 수 있는 명령어를 포함하는 소프트웨어로 구현될 수 있다. 기기(machine)는, 저장 매체로부터 저장된 명령어를 호출하고, 호출된 명령어에 따라 동작이 가능한 장치로서, 개시된 실시 예들의 전자 장치(700)를 포함할 수 있다.

[182] 이러한 명령어가 프로세서에 의해 실행될 경우, 프로세서가 직접, 또는 상기 프로세서의 제어 하에 다른 구성요소들을 이용하여 명령어에 해당하는 기능을 수행할 수 있다. 명령어는 컴파일러 또는 인터프리터에 의해 생성 또는 실행되는 코드를 포함할 수 있다. 예컨대, 저장매체에 저장된 명령어가 프로세서에 의해 실행됨으로써, 상술한 전자 장치의 제어방법이 실행될 수 있다.

[183] 기기로 읽을 수 있는 저장매체는, 비일시적(non-transitory) 저장매체의 형태로 제공될 수 있다. 여기서, '비일시적'은 저장매체가 신호(signal)를 포함하지 않으며 실재(tangible)한다는 것을 의미할 뿐 데이터가 저장매체에 반영구적 또는 임시적으로 저장됨을 구분하지 않는다.

[184] 일 실시 예에 따르면, 본 문서에 개시된 다양한 실시 예들에 따른 방법은 컴퓨터 프로그램 제품(computer program product)에 포함되어 제공될 수 있다. 컴퓨터 프로그램 제품은 상품으로서 판매자 및 구매자 간에 거래될 수 있다. 컴퓨터 프로그램 제품은 기기로 읽을 수 있는 저장 매체(예: compact disc read only memory (CD-ROM))의 형태로, 또는 어플리케이션 스토어(예: 플레이 스토어™, 앱스토어™)를 통해 온라인으로 배포될 수 있다. 온라인 배포의 경우에, 컴퓨터 프로그램 제품의 적어도 일부는 제조사의 서버, 어플리케이션 스토어의 서버, 또는 중계 서버의 메모리와 같은 저장 매체에 적어도 일시 저장되거나,

임시적으로 생성될 수 있다.

[185] 다양한 실시 예들에 따른 구성 요소(예: 모듈 또는 프로그램) 각각은 단수 또는 복수의 개체로 구성될 수 있으며, 전술한 해당 서브 구성 요소들 중 일부 서브 구성 요소가 생략되거나, 또는 다른 서브 구성 요소가 다양한 실시 예에 더 포함될 수 있다. 대체적으로 또는 추가적으로, 일부 구성 요소들(예: 모듈 또는 프로그램)은 하나의 개체로 통합되어, 통합되기 이전의 각각의 해당 구성 요소에 의해 수행되는 기능을 동일 또는 유사하게 수행할 수 있다. 다양한 실시 예들에 따른, 모듈, 프로그램 또는 다른 구성 요소에 의해 수행되는 동작들은 순차적, 병렬적, 반복적 또는 휴리스틱하게 실행되거나, 적어도 일부 동작이 다른 순서로 실행되거나, 생략되거나, 또는 다른 동작이 추가될 수 있다.

[186] 이상에서는 본 개시의 바람직한 실시 예에 대하여 도시하고 설명하였지만, 본 개시는 상술한 특정의 실시 예에 한정되지 아니하며, 청구범위에서 청구하는 본 개시의 요지를 벗어남이 없이 당해 개시에 속하는 기술분야에서 통상의 지식을 가진 자에 의해 다양한 변형실시가 가능한 것은 물론이고, 이러한 변형실시들은 본 개시의 기술적 사상이나 전망으로부터 개별적으로 이해되어서는 안될 것이다.

청구범위

[청구항 1] 서버리스(serverless) 플랫폼에 기반한 인공지능 서비스 제공 방법에 있어서,
인공지능 모델을 위한 라이브러리가 로딩된 복수의 컨테이너의 속성 및 모델 스토어에 등록된 복수의 인공지능 모델의 속성에 기초하여, 인공지능 모델이 로딩될 컨테이너를 판단하는 단계;
상기 컨테이너에 로딩된 라이브러리를 바탕으로 상기 컨테이너에 상기 인공지능 모델을 로딩하는 단계;
클라이언트로부터 인공지능 서비스에 대한 요청이 수신되면, 복수의 함수를 포함하는 데이터베이스로부터 상기 요청된 인공지능 서비스에 대응되는 함수를 획득하는 단계;
인공지능 모델이 로딩된 상기 복수의 컨테이너 중에서 상기 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단하는 단계;
상기 컨테이너에서 상기 컨테이너에 로딩된 라이브러리를 바탕으로 상기 획득된 함수를 실행하여, 상기 판단된 컨테이너에 로딩된 인공지능 모델로부터 상기 요청에 대한 데이터를 획득하는 단계; 및
상기 획득된 데이터를 상기 클라이언트로 전송하는 단계;를 포함하는, 인공지능 서비스 제공 방법.

[청구항 2] 제1항에 있어서,
상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는,
상기 복수의 컨테이너에 할당된 리소스 및 상기 복수의 인공지능 모델에 요구되는 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단하는, 인공지능 서비스 제공 방법.

[청구항 3] 제2항에 있어서,
상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는,
상기 복수의 컨테이너 각각에 할당된 리소스 및 상기 각 컨테이너에 로딩된 인공지능 모델에 요구되는 리소스에 기초하여 상기 각 컨테이너에서 사용 가능한 리소스를 판단하고, 상기 판단된 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단하는, 인공지능 서비스 제공 방법.

[청구항 4] 제1항에 있어서,
상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는,
상기 모델 스토어에 인공지능 모델이 등록되는 경우, 상기 복수의 컨테이너의 속성 및 상기 등록된 인공지능 모델의 속성에 기초하여 상기 등록된 인공지능 모델이 로딩될 컨테이너를 판단하는, 인공지능 서비스 제공 방법.

[청구항 5] 제1항에 있어서,

상기 컨테이너가 제1 상태인 상태에서 상기 획득된 함수를 실행하면,
상기 컨테이너의 상태를 제2 상태인 것으로 판단하는 단계; 및
상기 복수의 컨테이너 중 제1 상태인 컨테이너의 개수가 기 설정된 개수 미만이면, 새로운 컨테이너를 생성하는 단계;, 를 더 포함하는, 인공지능 서비스 제공 방법.

[청구항 6] 제1항에 있어서,
상기 획득된 함수를 실행한 컨테이너가 기설정된 시간 동안 상기 컨테이너에 포함된 모델을 이용하는 인공지능 서비스에 대한 요청을 수신하지 않은 경우, 상기 획득된 함수를 실행한 컨테이너를 킬(kill)하는 단계;를 더 포함하는, 인공지능 서비스 제공 방법.

[청구항 7] 제6항에 있어서,
상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는,
상기 킬 된 컨테이너를 제외한 나머지 컨테이너의 속성 및 상기 킬 된 컨테이너에 로딩되어 있던 인공지능 모델의 속성에 기초하여 상기 킬 된 컨테이너에 로딩되어 있던 인공지능 모델이 새롭게 로딩될 컨테이너를 판단하는, 인공지능 서비스 제공 방법.

[청구항 8] 제1항에 있어서,
상기 복수의 컨테이너는,
GPU(Graphic Processing Unit) 또는 CPU(Central Processing Unit)를 기반으로 상기 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수를 실행하는, 인공지능 서비스 제공 방법.

[청구항 9] 제8항에 있어서,
각 컨테이너에서의 상기 클라이언트 장치에서 요청된 인공지능 서비스에 대응되는 함수의 실행 시간에 대한 정보를 수집하는 단계;를 더 포함하고,
상기 인공지능 모델이 로딩될 컨테이너를 판단하는 단계는,
상기 수집된 정보를 바탕으로 상기 GPU 기반의 컨테이너 및 상기 CPU 기반의 컨테이너 중에서 상기 함수가 실행될 컨테이너를 판단하는,
인공지능 서비스 제공 방법.

[청구항 10] 서버리스(serverless) 플랫폼에 기반하여 인공지능 서비스를 제공하는 전자 장치에 있어서,
통신부;
복수의 함수를 포함하는 데이터베이스를 포함하는 메모리; 및
인공지능 모델을 위한 라이브러리가 로딩된 복수의 컨테이너의 속성 및 모델 스토어에 등록된 복수의 인공지능 모델의 속성에 기초하여,
인공지능 모델이 로딩될 컨테이너를 판단하고, 상기 컨테이너에 로딩된 라이브러리를 바탕으로 상기 컨테이너에 상기 인공지능 모델을 로딩하고, 상기 통신부를 통해 클라이언트 장치로부터 인공지능 서비스에 대한 요청이 수신되면, 상기 데이터베이스로부터 상기 요청된

인공지능 서비스에 대응되는 함수를 획득하고, 상기 복수의 컨테이너 중에서 상기 요청된 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단하고, 상기 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너에서 상기 컨테이너에 로딩된 라이브러리를 바탕으로 상기 획득된 함수를 실행하여, 상기 판단된 컨테이너에 로딩된 인공지능 모델로부터 상기 요청에 대한 데이터를 획득하고, 상기 획득된 데이터를 상기 통신부를 통해 상기 클라이언트로 전송하는 프로세서;를 포함하는, 전자 장치.

[청구항 11] 제10항에 있어서,
상기 프로세서는,
상기 복수의 컨테이너에 할당된 리소스 및 상기 복수의 인공지능 모델에 요구되는 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단하는, 전자 장치.

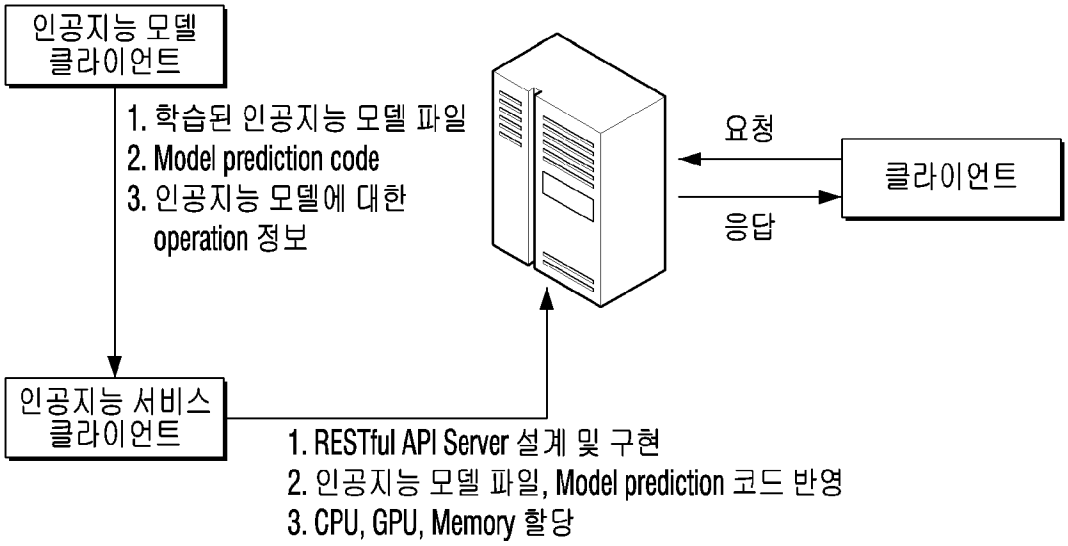
[청구항 12] 제11항에 있어서,
상기 프로세서는,
상기 복수의 컨테이너 각각에 할당된 리소스 및 상기 각 컨테이너에 로딩된 인공지능 모델에 요구되는 리소스에 기초하여 상기 각 컨테이너에서 사용 가능한 리소스를 판단하고, 상기 판단된 리소스에 기초하여 상기 인공지능 모델이 로딩될 상기 컨테이너를 판단하는, 전자 장치.

[청구항 13] 제10항에 있어서,
상기 프로세서는,
상기 모델 스토어에 인공지능 모델이 등록되는 경우, 상기 복수의 컨테이너의 속성 및 상기 등록된 인공지능 모델의 속성에 기초하여 상기 등록된 인공지능 모델이 로딩될 컨테이너를 판단하는, 전자 장치.

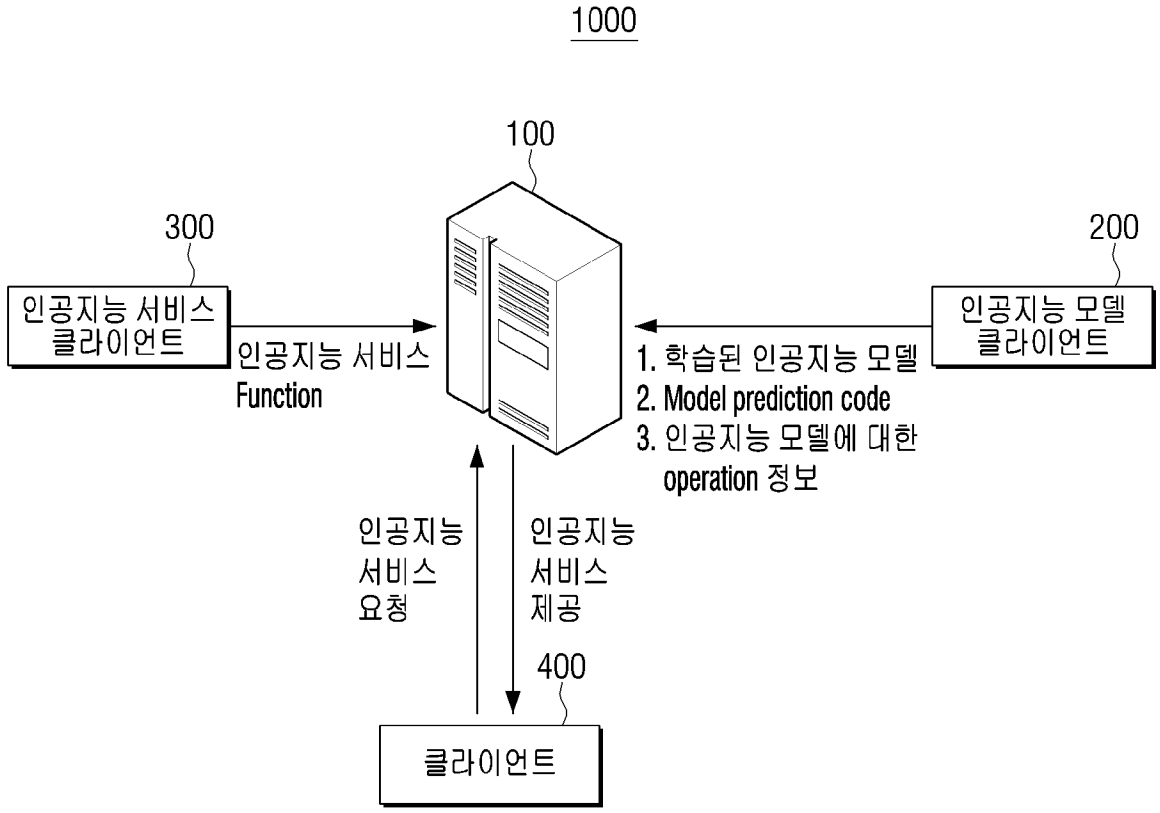
[청구항 14] 제10항에 있어서,
상기 프로세서는,
상기 컨테이너가 제1 상태인 상태에서 상기 획득된 함수를 실행하면, 상기 컨테이너의 상태가 제2 상태인 것으로 판단하고,
상기 복수의 컨테이너 중 제1 상태인 컨테이너의 개수가 기 설정된 수 미만이면, 새로운 컨테이너를 생성하는, 전자 장치.

[청구항 15] 제10항에 있어서,
상기 프로세서는,
상기 획득된 함수를 실행한 컨테이너가 기 설정된 시간 동안 상기 컨테이너에 포함된 모델을 이용하는 인공지능 서비스에 대한 요청을 수신하지 않은 경우, 상기 획득된 함수를 실행한 컨테이너를 킬(kill)하는, 전자 장치.

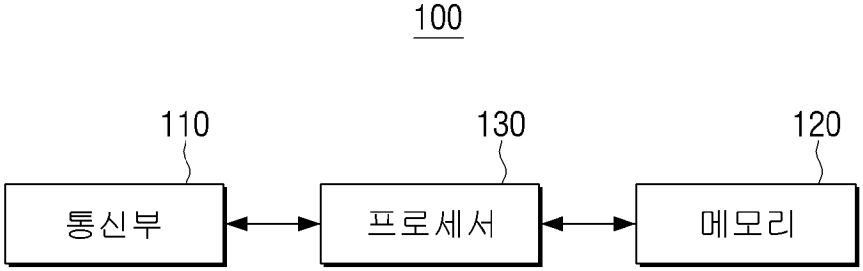
[도1]


인공지능 모델
클라이언트
1. 학습된 인공지능 모델 파일
2. Model prediction code
3. 인공지능 모델에 대한
operation 정보
요청
응답
클라이언트
인공지능 서비스
클라이언트
1. RESTful API Server 설계 및 구현
2. 인공지능 모델 파일, Model prediction 코드 반영
3. CPU, GPU, Memory 할당

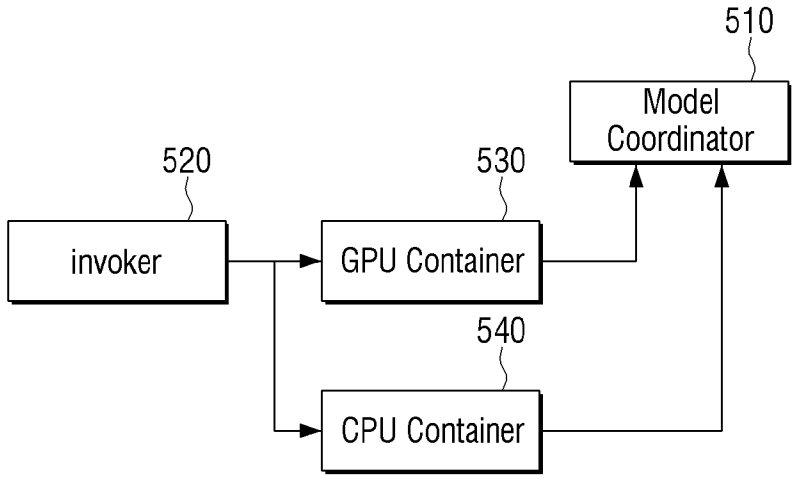
[도2]


1000
100
300
200
인공지능 서비스
클라이언트
인공지능 서비스
Function
인공지능 모델
클라이언트
1. 학습된 인공지능 모델
2. Model prediction code
3. 인공지능 모델에 대한
operation 정보
인공지능
서비스
요청
인공지능
서비스
제공
400
클라이언트

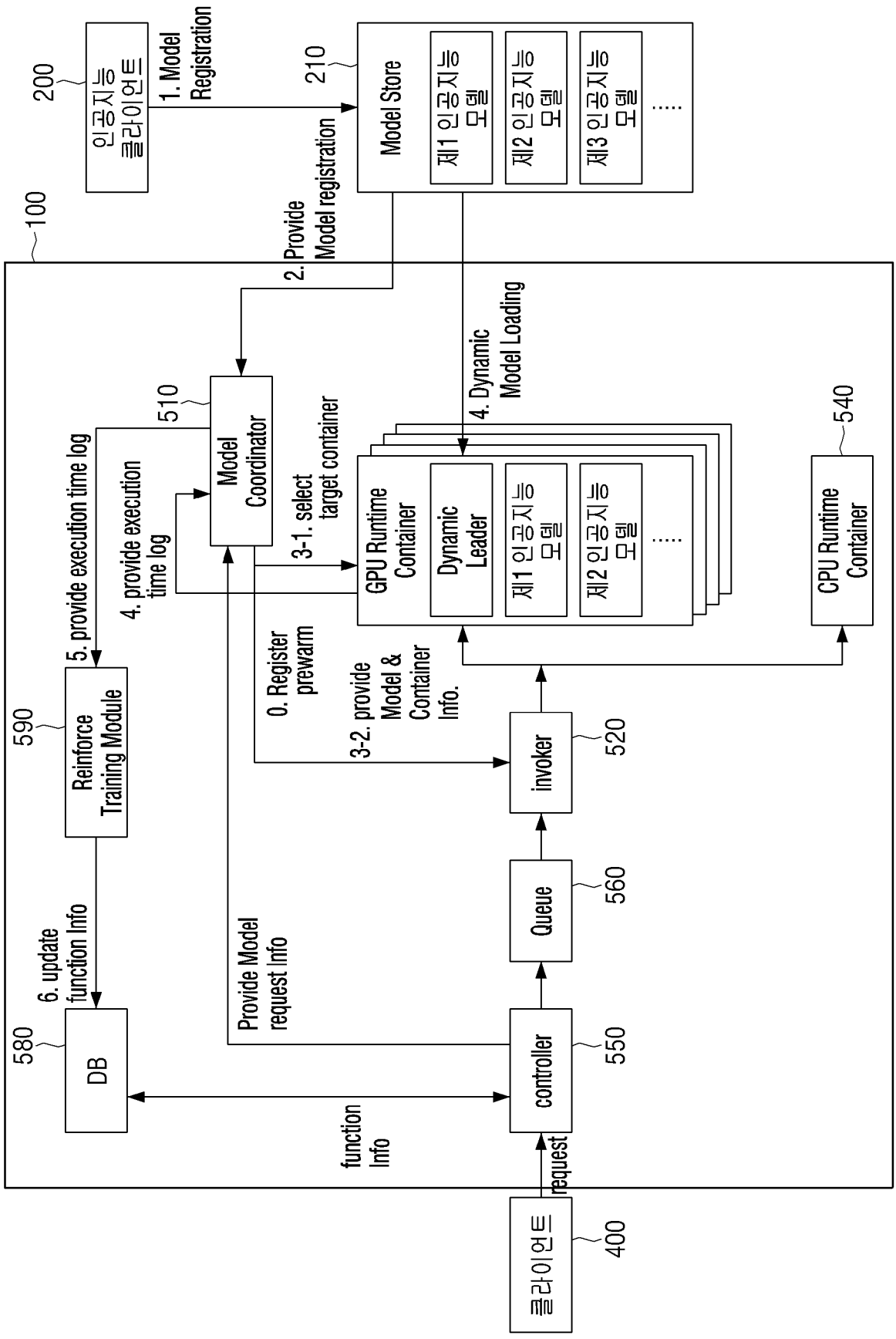
[도3]


100
110
130
120
통신부
프로세서
메모리

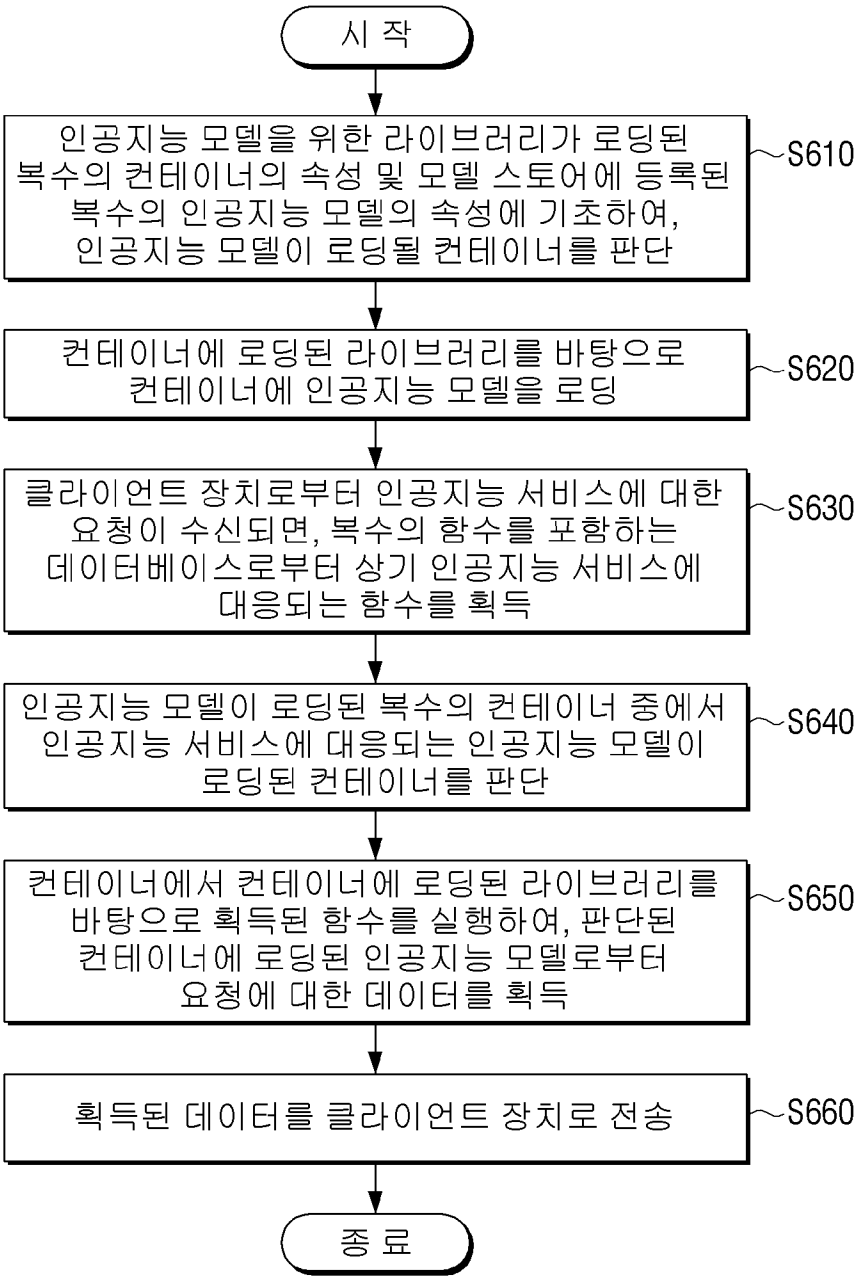
[도4]


510
Model Coordinator
520
530
invoker
GPU Container
540
CPU Container

[도5]


200
인공지능 클라이언트
1. Model Registration
210
Model Store
제1 인공지능 모델
제2 인공지능 모델
제3 인공지능 모델
.....
100
2. Provide Model registration
510
Model Coordinator
4. Dynamic Model Loading
GPU Runtime Container
Dynamic Leader
제1 인공지능 모델
제2 인공지능 모델
.....
540
CPU Runtime Container
3-1. select target container
4. provide execution time log
5. provide execution time log
0. Register prewarm
3-2. provide Model & Container Info.
590
Reinforce Training Module
520
invoker
560
Queue
6. update function Info
580
DB
Provide Model request Info
550
controller
function Info
request
400
클라이언트

[도6]


시 작
인공지능 모델을 위한 라이브러리가 로딩된 복수의 컨테이너의 속성 및 모델 스토어에 등록된 복수의 인공지능 모델의 속성에 기초하여, 인공지능 모델이 로딩될 컨테이너를 판단
S610
컨테이너에 로딩된 라이브러리를 바탕으로 컨테이너에 인공지능 모델을 로딩
S620
클라이언트 장치로부터 인공지능 서비스에 대한 요청이 수신되면, 복수의 함수를 포함하는 데이터베이스로부터 상기 인공지능 서비스에 대응되는 함수를 획득
S630
인공지능 모델이 로딩된 복수의 컨테이너 중에서 인공지능 서비스에 대응되는 인공지능 모델이 로딩된 컨테이너를 판단
S640
컨테이너에서 컨테이너에 로딩된 라이브러리를 바탕으로 획득된 함수를 실행하여, 판단된 컨테이너에 로딩된 인공지능 모델로부터 요청에 대한 데이터를 획득
S650
획득된 데이터를 클라이언트 장치로 전송
S660
종 료

INTERNATIONAL SEARCH REPORT

International application No.
PCT/KR2020/000230

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/50(2006.01)i, G06F 9/48(2006.01)i, G06F 9/445(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 9/50; G06F 9/455; G06N 3/08; G06N 99/00; G06F 9/48; G06F 9/445

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Korean utility models and applications for utility models: IPC as above
Japanese utility models and applications for utility models: IPC as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS (KIPO internal) & Keywords: serverless, artificial intelligent model, container, service, function

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	ISHAKIAN, Vatche et al. Serving deep learning models in a serverless platform. In: 2018 IEEE International Conference on Cloud Engineering (IC2E). DOI: 10.1109/IC2E 2018.00052. 17 April 2018 See sections II, V.	1-15
A	US 2019-0050756 A1 (AMAZON TECHNOLOGIES, INC.) 14 February 2019 See paragraphs [0034], [0046]-[0047]; claim 23; and figures 1, 3.	1-15
A	US 2017-0213156 A1 (BONSAI AI, INC.) 27 July 2017 See paragraphs [0023], [0027], [0031]; and claim 1.	1-15
A	KR 10-2017-0085072 A (AMAZON TECHNOLOGIES, INC.) 21 July 2017 See paragraphs [0019]-[0024]; and figure 1.	1-15
A	장경수 등. TensorFlow Serving 서비스를 지원하는 고성능 GPU 기반 컨테이너 클라우드 시스템. 2017년 춘계학술발표대회 논문집. vol. 24, no. 2, pages 386-388, November 2017 (JANG, Kyung-soo et al. A Study on High Performance GPU Based Container Cloud System Supporting TensorFlow Serving Deployment Service. 2017 Proceedings of Spring Conference.) See pages 386-388.	1-15

☐ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
17 APRIL 2020 (17.04.2020)	**17 APRIL 2020 (17.04.2020)**
Name and mailing address of the ISA/KR Korean Intellectual Property Office Government Complex Daejeon Building 4, 189, Cheongsa-ro, Seo-gu, Daejeon, 35208, Republic of Korea Facsimile No. +82-42-481-8578	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (January 2015)

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/KR2020/000230

Patent document cited in search report	Publication date	Patent family member	Publication date
US 2019-0050756 A1	14/02/2019	US 10102480 B2	16/10/2018
		US 2015-0379424 A1	31/12/2015
US 2017-0213156 A1	27/07/2017	CN 109496320 A	19/03/2019
		CN 109564505 A	02/04/2019
		CN 110168495 A	23/08/2019
		EP 3408739 A1	05/12/2018
		EP 3408739 A4	02/10/2019
		EP 3408750 A1	05/12/2018
		EP 3408750 A4	25/09/2019
		EP 3408800 A1	05/12/2018
		EP 3408800 A4	18/09/2019
		US 2017-0213126 A1	27/07/2017
		US 2017-0213128 A1	27/07/2017
		US 2017-0213131 A1	27/07/2017
		US 2017-0213132 A1	27/07/2017
		US 2017-0213154 A1	27/07/2017
		US 2017-0213155 A1	27/07/2017
		US 2018-0293463 A1	11/10/2018
		US 2018-0293498 A1	11/10/2018
		US 2018-0293517 A1	11/10/2018
		US 2018-0307945 A1	25/10/2018
		US 2018-0357047 A1	13/12/2018
		US 2018-0357152 A1	13/12/2018
		US 2018-0357543 A1	13/12/2018
		US 2018-0357552 A1	13/12/2018
		WO 2017-132572 A1	03/08/2017
		WO 2017-132584 A1	03/08/2017
		WO 2017-132590 A1	03/08/2017
		WO 2018-236674 A1	27/12/2018
KR 10-2017-0085072 A	21/07/2017	AU 2017-346530 A1	01/06/2017
		AU 2019-204805 A1	25/07/2019
		CN 107111519 A	29/08/2017
		EP 3218808 A1	20/09/2017
		JP 2017-538204 A	21/12/2017
		JP 2019-149192 A	05/09/2019
		JP 6522750 B2	29/05/2019
		KR 10-2019-0020843 A	04/03/2019
		KR 10-2042988 B1	11/11/2019
		RU 2018130629 A	15/03/2019
		RU 2666475 C1	07/09/2018
		RU 2704734 C2	30/10/2019
		SG 11201703680 A	29/06/2017
		US 2016-0162320 A1	09/06/2016
		US 2019-0108049 A1	11/04/2019
		US 9256467 B1	09/02/2016
		US 9996380 B2	12/06/2018
		WO 2016-077367 A1	19/05/2016

국 제 조 사 보 고 서

국제출원번호
PCT/KR2020/000230

A. 발명이 속하는 기술분류(국제특허분류(IPC))

G06F 9/50(2006.01)i, G06F 9/48(2006.01)i, G06F 9/445(2006.01)i

B. 조사된 분야

조사된 최소문헌(국제특허분류를 기재)
G06F 9/50; G06F 9/455; G06N 3/08; G06N 99/00; G06F 9/48; G06F 9/445

조사된 기술분야에 속하는 최소문헌 이외의 문헌
한국등록실용신안공보 및 한국공개실용신안공보: 조사된 최소문헌란에 기재된 IPC
일본등록실용신안공보 및 일본공개실용신안공보: 조사된 최소문헌란에 기재된 IPC

국제조사에 이용된 전산 데이터베이스(데이터베이스의 명칭 및 검색어(해당하는 경우))
eKOMPASS(특허청 내부 검색시스템) & 키워드: 서버리스(serverless), 인공지능 모델(artificial intelligent model), 컨테이너(container), 서비스(service), 함수(function)

C. 관련 문헌

카테고리*	인용문헌명 및 관련 구절(해당하는 경우)의 기재	관련 청구항
A	VATCHE ISHAKIAN 등, `Serving deep learning models in a serverless platform`, In: 2018 IEEE International Conference on Cloud Engineering (IC2E), DOI: 10.1109/IC2E 2018.00052, 2018.04.17 섹션 II, V	1-15
A	US 2019-0050756 A1 (AMAZON TECHNOLOGIES, INC.) 2019.02.14 단락 [0034], [0046]-[0047]; 청구항 23; 및 도면 1, 3	1-15
A	US 2017-0213156 A1 (BONSAI AI, INC.) 2017.07.27 단락 [0023], [0027], [0031]; 및 청구항 1	1-15
A	KR 10-2017-0085072 A (아마존 테크놀로지스, 인크.) 2017.07.21 단락 [0019]-[0024]; 및 도면 1	1-15
A	장경수 등, `TensorFlow Serving 서비스를 지원하는 고성능 GPU 기반 컨테이너 클라우드 시스템`, 2017년 춘계학술발표대회 논문집 제24권 제2호, 페이지 386-388, 2017.11 페이지 386-388	1-15

☐ 추가 문헌이 C(계속)에 기재되어 있습니다.

☒ 대응특허에 관한 별지를 참조하십시오.

* 인용된 문헌의 특별 카테고리:

"A" 특별히 관련이 없는 것으로 보이는 일반적인 기술수준을 정의한 문헌

"D" 본 국제출원에서 출원인이 인용한 문헌

"E" 국제출원일보다 빠른 출원일 또는 우선일을 가지나 국제출원일 이후에 공개된 선출원 또는 특허 문헌

"L" 우선권 주장에 의문을 제기하는 문헌 또는 다른 인용문헌의 공개일 또는 다른 특별한 이유(이유를 명시)를 밝히기 위하여 인용된 문헌

"O" 구두 개시, 사용, 전시 또는 기타 수단을 언급하고 있는 문헌

"P" 우선일 이후에 공개되었으나 국제출원일 이전에 공개된 문헌

"T" 국제출원일 또는 우선일 후에 공개된 문헌으로, 출원과 상충하지 않으며 발명의 기초가 되는 원리나 이론을 이해하기 위해 인용된 문헌

"X" 특별한 관련이 있는 문헌. 해당 문헌 하나만으로 청구된 발명의 신규성 또는 진보성이 없는 것으로 본다.

"Y" 특별한 관련이 있는 문헌. 해당 문헌이 하나 이상의 다른 문헌과 조합하는 경우로 그 조합이 당업자에게 자명한 경우 청구된 발명은 진보성이 없는 것으로 본다.

"&" 동일한 대응특허문헌에 속하는 문헌

국제조사의 실제 완료일	국제조사보고서 발송일
2020년 04월 17일 (17.04.2020)	2020년 04월 17일 (17.04.2020)

ISA/KR의 명칭 및 우편주소
대한민국 특허청
(35208) 대전광역시 서구 청사로 189, 4동 (둔산동, 정부대전청사)
팩스 번호 +82-42-481-8578

심사관
김성희
전화번호 +82-42-481-3516

국제조사보고서
대응특허에 관한 정보

국제출원번호
PCT/KR2020/000230

국제조사보고서에서 인용된 특허문헌	공개일	대응특허문헌	공개일
US 2019-0050756 A1	2019/02/14	US 10102480 B2	2018/10/16
		US 2015-0379424 A1	2015/12/31
US 2017-0213156 A1	2017/07/27	CN 109496320 A	2019/03/19
		CN 109564505 A	2019/04/02
		CN 110168495 A	2019/08/23
		EP 3408739 A1	2018/12/05
		EP 3408739 A4	2019/10/02
		EP 3408750 A1	2018/12/05
		EP 3408750 A4	2019/09/25
		EP 3408800 A1	2018/12/05
		EP 3408800 A4	2019/09/18
		US 2017-0213126 A1	2017/07/27
		US 2017-0213128 A1	2017/07/27
		US 2017-0213131 A1	2017/07/27
		US 2017-0213132 A1	2017/07/27
		US 2017-0213154 A1	2017/07/27
		US 2017-0213155 A1	2017/07/27
		US 2018-0293463 A1	2018/10/11
		US 2018-0293498 A1	2018/10/11
		US 2018-0293517 A1	2018/10/11
		US 2018-0307945 A1	2018/10/25
		US 2018-0357047 A1	2018/12/13
		US 2018-0357152 A1	2018/12/13
		US 2018-0357543 A1	2018/12/13
		US 2018-0357552 A1	2018/12/13
		WO 2017-132572 A1	2017/08/03
		WO 2017-132584 A1	2017/08/03
		WO 2017-132590 A1	2017/08/03
		WO 2018-236674 A1	2018/12/27
KR 10-2017-0085072 A	2017/07/21	AU 2017-346530 A1	2017/06/01
		AU 2019-204805 A1	2019/07/25
		CN 107111519 A	2017/08/29
		EP 3218808 A1	2017/09/20
		JP 2017-538204 A	2017/12/21
		JP 2019-149192 A	2019/09/05
		JP 6522750 B2	2019/05/29
		KR 10-2019-0020843 A	2019/03/04
		KR 10-2042988 B1	2019/11/11
		RU 2018130629 A	2019/03/15
		RU 2666475 C1	2018/09/07
		RU 2704734 C2	2019/10/30
		SG 11201703680 A	2017/06/29
		US 2016-0162320 A1	2016/06/09
		US 2019-0108049 A1	2019/04/11
		US 9256467 B1	2016/02/09
		US 9996380 B2	2018/06/12
		WO 2016-077367 A1	2016/05/19