

(19) 대한민국특허청(KR)
(12) 공개특허공보(A)(11) 공개번호 10-2024-0031836
(43) 공개일자 2024년03월08일

- (51) 국제특허분류(Int. Cl.)
G06F 21/60 (2013.01) G06F 16/16 (2019.01)
G06F 16/176 (2019.01) G06F 40/149 (2020.01)
G06N 3/08 (2023.01)
- (52) CPC특허분류
G06F 21/60 (2013.01)
G06F 16/164 (2019.01)
- (21) 출원번호 10-2022-0137104
(22) 출원일자 2022년10월24일
심사청구일자 2022년10월24일
- (30) 우선권주장
1020220109259 2022년08월30일 대한민국(KR)
- (71) 출원인
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
- (72) 발명자
김준호
경기도 성남시 중원구 광명로323번길 20-10, 1동 301호 (금광동, 금광동 3573 공동주택 (김상윤))
박정흠
서울시 마포구 마포대로 195 408동 2205호
이인혁
부산광역시 기장군 기장읍 차성로365번길 63, 10 3동 503호(리츠빌리지)
- (74) 대리인
특허법인위솔

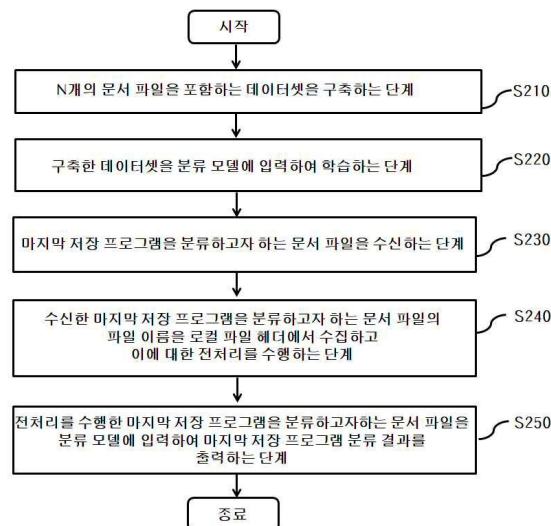
전체 청구항 수 : 총 11 항

(54) 발명의 명칭 문서 파일의 마지막 저장 프로그램을 분류하는 방법 및 이를 위한 장치

(57) 요약

본 발명의 일 실시 예에 따른 프로세서 및 메모리를 포함하는 장치가 문서 파일의 마지막 저장 프로그램을 분류하는 방법은 (a) N(N은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 단계, (b) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 단계, (c) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 단계, (d) 상기 수신한 마지막 저장 프로그램을 분류하고자 하는 문서 파일에 대한 전처리를 수행하는 단계 및 (e) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 단계를 포함하며, 상기 N개의 파일은, OOXML(Office Open XML) 기반 문서 파일이다.

대표도 - 도2



(52) CPC특허분류

G06F 16/176 (2019.01)

G06F 40/149 (2020.01)

G06N 3/08 (2023.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711170650
과제번호	2022-0-00281-001
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	정보보호핵심원천기술개발(정진기금)
연구과제명	인공지능 기술 활용 디지털증거 분석 기법 개발
기 여 율	1/1
과제수행기관명	동국대학교산학협력단
연구기간	2022.04.01 ~ 2022.12.31

명세서

청구범위

청구항 1

프로세서 및 메모리를 포함하는 장치가 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서,

(a) N (N 은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 단계;

(b) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 단계;

(c) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 단계;

(d) 상기 수신한 마지막 저장 프로그램을 분류하고자하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 단계; 및

(e) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 단계;

를 포함하며,

상기 N 개의 문서 파일은,

OOXML(Office Open XML) 기반 문서 파일인,

문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 2

제1항에 있어서,

상기 N 개의 문서 파일은,

N 개의 워드(Word) 문서 파일, N 개의 파워포인트(Powerpoint) 문서 파일 및 N 개의 엑셀(Excel) 문서 파일 중 어느 하나 이상인,

문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 3

제1항에 있어서,

상기 (a) 단계는,

(a-1) 상기 N 개의 문서 파일을 수집하는 단계;

(a-2) 상기 수집한 N 개의 문서 파일 각각을 M (M 은 2 이상의 정수)개의 문서 생성/수정 프로그램 별로 수정하여 저장하는 단계;

(a-3) 상기 M 개의 문서 생성/수정 프로그램 별로 수정하여 저장한 N 개의 파일 각각의 로컬 파일 헤더(Local File Header) 내 파일 이름(File Name)을 상기 M 개의 문서 생성/수정 프로그램 별로 순서대로 수집하는 단계;

(a-4) 상기 M 개의 문서 생성/수정 프로그램 별로 상기 수집한 파일 이름 각각에 대한 전처리를 수행하는 단계; 및

(a-5) 상기 M 개의 문서 생성/수정 프로그램 별로 파일 이름이 전처리된 N 개의 문서 파일이 대응된 데이터셋을 구축하는 단계;

를 포함하는 문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 4

제3항에 있어서,

상기 (a-3) 단계는,

(a-3-1) 상기 M개의 문서 생성/수정 프로그램 별로 수정하여 저장한 N개의 파일 각각의 로컬 파일 헤더(Local File Header) 내 파일 이름 길이(File Name Length)를 순서대로 확인하는 단계; 및

(a-3-2) 상기 파일 이름 길이를 확인한 N개의 파일 각각의 로컬 파일 헤더 내 파일 이름을 상기 M개의 문서 생성/수정 프로그램 별로 순서대로 수집하는 단계;

를 포함하는 문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 5

제3항에 있어서,

상기 (a-4) 단계는,

(a-4-1) 상기 수집한 파일 이름 각각이 동일한 파일 이름과 서로 다른 숫자들이 조합된 경우, 가장 처음 확인된 파일 이름만 숫자를 제거하고, 그 이후에 확인된 나머지 파일 이름은 전부 삭제하는 전처리를 수행하는 단계;

를 포함하는 문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 6

제3항에 있어서,

상기 (a-4) 단계는,

(a-4-2) 상기 수집한 파일 이름 각각이 대문자 및 소문자를 혼용하여 조합된 경우, 상기 파일 이름 각각을 전부 소문자로 통일시키는 전처리를 수행하는 단계;

를 포함하는 문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 7

제3항에 있어서,

상기 (a-4) 단계는,

(a-4-3) 상기 수집한 파일 이름 각각이 불용어를 포함하여 조합된 경우, 상기 각각의 파일 이름에서 불용어를 삭제하는 전처리를 수행하는 단계;

를 포함하는 문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 8

제1항에 있어서,

상기 분류 모델은,

장단기 메모리인 LSTM(Long Short Term Memory) 네트워크 구조를 갖는 분류 모델인,

문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 9

제8항에 있어서,

상기 분류 모델의 하이퍼파라미터(Hyperparameter)는,

LSTM Cell은 30, Dropout rate는 0.2, Activation Function은 Softmax, Optimizer는 Adam, Loss Function은 Categorical Crossentropy, Epoch는 10 및 Batch Size는 64인,

문서 파일의 마지막 저장 프로그램을 분류하는 방법.

청구항 10

하나 이상의 프로세서;

네트워크 인터페이스;

상기 프로세서에 의해 수행되는 컴퓨터 프로그램을 로드(Load)하는 메모리; 및

대용량 네트워크 데이터 및 상기 컴퓨터 프로그램을 저장하는 스토리지를 포함하되,

상기 컴퓨터 프로그램은 상기 하나 이상의 프로세서에 의해,

(A) N (N 은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 오퍼레이션;

(B) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 오퍼레이션;

(C) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 오퍼레이션;

(D) 상기 수신한 마지막 저장 프로그램을 분류하고자하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 오퍼레이션; 및

(E) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 오퍼레이션;

을 실행하며,

상기 N 개의 문서 파일은,

OOXML(Office Open XML) 기반 문서 파일인,

문서 파일의 마지막 저장 프로그램을 분류하는 장치.

청구항 11

컴퓨팅 장치와 결합하여,

(AA) N (N 은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 단계;

(BB) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 단계;

(CC) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 단계;

(DD) 상기 수신한 마지막 저장 프로그램을 분류하고자하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 단계; 및

(EE) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 단계;

를 실행하며,

상기 N 개의 문서 파일은,

OOXML(Office Open XML) 기반 문서 파일인,

컴퓨터로 판독 가능한 매체에 저장된 컴퓨터 프로그램.

발명의 설명

기술 분야

[0001] 본 발명은 문서 파일의 마지막 저장 프로그램을 분류하는 방법 및 이를 위한 장치에 관한 것이다. 보다 자세하게는 문서 파일의 내부 데이터를 분석하여 디지털포렌식 활동을 위한 마지막 저장 프로그램을 분류할 수 있는 방법 및 장치에 관한 것이다.

배경 기술

- [0002] 프로그램 중에서 특정한 업무를 해결하기 위한 목적을 가지고 만들어진 프로그램을 일컬어 응용프로그램(Application Program)이라하며, 기술의 발전에 따라 다양한 응용프로그램이 생겨나고 있고, 이러한 응용프로그램은 무수히 많은 종류의 파일을 생성해내고 있다.
- [0003] 한편, 컴퓨터가 보급된 이후 종래 수작업으로 이루어지던 업무 대부분이 컴퓨터 상에서 이루어지게 되며, 응용프로그램 중, 컴퓨터에 설치되어 문서 파일을 다루는 문서 생성/수정 프로그램들이 필연적으로 다양하게 출현하게 되었으며, 마이크로소프트 오피스, 리브레 오피스, 한컴 오피스, WPS 오피스 등을 대표적으로 들 수 있다.
- [0004] 더 나아가 인터넷 기술의 발달로 인해 최근에는 컴퓨터에 설치되는 문서 생성/수정 프로그램들뿐만 아니라 웹(Web) 상에서 문서를 바로 생성하거나 수정할 수 있는 클라우드(Cloud) 기반 응용프로그램들도 다수 생겨난바, 구글 워크스페이스, 한컴 스페이스, 네이버 오피스, A1 오피스 등을 대표적으로 들 수 있다.
- [0005] 한편, 디지털포렌식이란 디지털 증거물을 분석하여 수사에 활용하고, 디지털 증거물의 증거 능력을 향상시키기 위한 과학 수사 기법을 의미하는바, 디지털 증거물 중 문서 파일이 많은 비중을 차지하고 있으며, 중요한 증거가 될 수 있는 문서 파일의 마지막 수정 여부가 쟁점이 되었을 때, 해당 문서 파일이 해당 컴퓨터에서 저장되었는지 파악하기가 매우 어렵다는 문제점이 있다.
- [0006] 그러나 해당 문서 파일의 마지막 저장 프로그램을 파악하고, 해당 컴퓨터에 설치된 프로그램 목록들의 설치 시간들과 문서 파일의 마지막 저장 시간을 비교한다면 해당 문서 파일이 해당 컴퓨터에서 저장되었는지 파악하는데 도움을 줄 수 있을 것인바, 문서 파일의 마지막 저장 프로그램을 파악하기 위한 다양한 연구가 계속되고 있다. 본 발명은 이에 관한 것이다.

선행기술문헌

특허문헌

- [0007] (특허문헌 0001) 대한민국 공개특허공보 제 10-2014-0000369호(2014.01.03)

발명의 내용

해결하려는 과제

- [0008] 본 발명이 해결하고자 하는 기술적 과제는 디지털포렌식 활동에 있어서 문서 파일의 근원지를 파악할 때 용이하게 활용될 수 있는 문서 파일의 마지막 저장 프로그램을 분류하는 방법 및 이를 위한 장치를 제공하는 것이다.
- [0009] 본 발명이 해결하고자 하는 또 다른 기술적 과제는 인공지능 모델의 학습 데이터의 크기가 크지 않고, 문서 파일의 마지막 저장 프로그램 분류시 활용하는 데이터의 양을 적게 함으로써 수사관이 현장에서 활용할 수 있는 이동형 디지털포렌식 분석 시스템에 탑재할 수 있는 문서 파일의 마지막 저장 프로그램을 분류하는 방법 및 이를 위한 장치를 제공하는 것이다.
- [0010] 본 발명의 기술적 과제들은 이상에서 언급한 기술적 과제들로 제한되지 않으며, 언급되지 않은 또 다른 기술적 과제들은 아래의 기재로부터 통상의 기술자에게 명확하게 이해될 수 있을 것이다.

과제의 해결 수단

- [0011] 상기 기술적 과제를 달성하기 위한 본 발명의 일 실시 예에 따른 프로세서 및 메모리를 포함하는 장치가 문서 파일의 마지막 저장 프로그램을 분류하는 방법은 (a) N(N은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 단계, (b) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 단계, (c) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 단계, (d) 상기 수신한 마지막 저장 프로그램을 분류하고자 하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 단계 및 (e) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 단계를 포함하며, 상기 N개의 파일은, OOXML(Office Open XML) 기반 문서 파일이다.
- [0012] 일 실시 예에 따르면, 상기 N개의 문서 파일은, N개의 워드(Word) 문서 파일, N개의 파워포인트(Powerpoint) 문서 파일 및 N개의 엑셀(Excel) 문서 파일 중 어느 하나 이상일 수 있다.
- [0013] 일 실시 예에 따르면, 상기 (a) 단계는, (a-1) 상기 N개의 문서 파일을 수집하는 단계, (a-2) 상기 수집한 N개

의 문서 파일 각각을 M(M은 2 이상의 정수)개의 문서 생성/수정 프로그램 별로 수정하여 저장하는 단계, (a-3) 상기 M개의 문서 생성/수정 프로그램 별로 수정하여 저장한 N개의 파일 각각의 로컬 파일 헤더(Local File Header) 내 파일 이름(File Name)을 상기 M개의 문서 생성/수정 프로그램 별로 순서대로 수집하는 단계, (a-4) 상기 M개의 문서 생성/수정 프로그램 별로 상기 수집한 파일 이름 각각에 대한 전처리를 수행하는 단계 및 (a-5) 상기 M개의 문서 생성/수정 프로그램 별로 파일 이름이 전처리된 N개의 문서 파일이 대응된 데이터셋을 구축하는 단계를 포함할 수 있다.

[0014] 일 실시 예에 따르면, 상기 (a-3) 단계는, (a-3-1) 상기 M개의 문서 생성/수정 프로그램 별로 수정하여 저장한 N개의 파일 각각의 로컬 파일 헤더(Local File Header) 내 파일 이름 길이(File Name Length)를 순서대로 확인하는 단계 및 (a-3-2) 상기 파일 이름 길이를 확인한 N개의 파일 각각의 로컬 파일 헤더 내 파일 이름을 상기 M개의 문서 생성/수정 프로그램 별로 순서대로 수집하는 단계를 포함할 수 있다.

[0015] 일 실시 예에 따르면, 상기 (a-4) 단계는, (a-4-1) 상기 수집한 파일 이름 각각이 동일한 파일 이름과 서로 다른 숫자들이 조합된 경우, 가장 처음 확인된 파일 이름만 숫자를 제거하고, 그 이후에 확인된 나머지 파일 이름은 전부 삭제하는 전처리를 수행하는 단계를 포함할 수 있다.

[0016] 일 실시 예에 따르면, 상기 (a-4) 단계는, (a-4-2) 상기 수집한 파일 이름 각각이 대문자 및 소문자를 혼용하여 조합된 경우, 상기 파일 이름 각각을 전부 소문자로 통일시키는 전처리를 수행하는 단계를 포함할 수 있다.

[0017] 일 실시 예에 따르면, 상기 (a-4) 단계는, (a-4-3) 상기 수집한 파일 이름 각각이 불용어를 포함하여 조합된 경우, 상기 각각의 파일 이름에서 불용어를 삭제하는 전처리를 수행하는 단계를 포함할 수 있다.

[0018] 일 실시 예에 따르면, 상기 분류 모델은, 장단기 메모리인 LSTM(Long Short Term Memory) 네트워크 구조를 갖는 분류 모델일 수 있다.

[0019] 일 실시 예에 따르면, 상기 분류 모델의 하이퍼파라미터(Hyperparameter)는, LSTM Cell은 30, Dropout rate는 0.2, Activation Function은 Softmax, Optimizer는 Adam, Loss Function은 Categorical Crossentropy, Epoch는 10 및 Batch Size는 64일 수 있다.

[0020] 상기 기술적 과제를 달성하기 위한 본 발명의 또 다른 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치는 하나 이상의 프로세서, 네트워크 인터페이스, 상기 프로세서에 의해 수행되는 컴퓨터 프로그램을 로드(Load)하는 메모리 및 대용량 네트워크 데이터 및 상기 컴퓨터 프로그램을 저장하는 스토리지를 포함하되, 상기 컴퓨터 프로그램은 상기 하나 이상의 프로세서에 의해, (A) N(N은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 오퍼레이션, (B) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 오퍼레이션, (C) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 오퍼레이션, (D) 상기 수신한 마지막 저장 프로그램을 분류하고자 하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 오퍼레이션 및 (E) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 오퍼레이션을 실행하며, 상기 N개의 문서 파일은, OOXML(Office Open XML) 기반 문서 파일이다.

[0021] 상기 기술적 과제를 달성하기 위한 본 발명의 또 다른 실시 예에 따른 매체에 저장된 컴퓨터 프로그램은 컴퓨팅 장치와 결합하여, (AA) N(N은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 단계, (BB) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 단계, (CC) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 단계, (DD) 상기 수신한 마지막 저장 프로그램을 분류하고자 하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 단계 및 (EE) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 단계를 실행하며, 상기 N개의 문서 파일은, OOXML(Office Open XML) 기반 문서 파일이다.

발명의 효과

[0022] 상기와 같은 본 발명에 따르면, 데이터셋을 구축하여 분류 모델에 학습시킨 이후, 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하여 분류 모델에 입력하기만 하면 분류 결과가 손쉽게 출력되는바, 과제는 디지털포렌식 활동에 있어서 문서 파일의 근원지를 파악할 때 용이하게 활용할 수 있다는 효과가 있다.

[0023] 또한, 500개 정도의 문서 파일을 통해 분류 모델의 학습이 가능하기에 학습에 필요한 데이터의 크기가 크지 않고, 문서 파일의 마지막 저장 프로그램 분류시 활용하는 데이터의 양까지 적게할 수 있는바, 수사관이 현장에서 활용할 수 있는 이동형 디지털포렌식 분석 시스템에 손쉽게 탑재함으로써 편의성을 향상시킬 수 있다는 효과가

있다.

[0024] 또한, 분류 모델의 학습이 신속하게 이루어질 수 있으며, 사용하면 할수록 문서 파일의 마지막 저장 프로그램 분류 결과를 지속적으로 학습하기 때문에 분류의 정확도가 향상될 수 있음과 동시에 분류 결과의 출력 역시 신속하게 이루어질 수 있다

[0025] 는 효과가 있다.

[0026] 본 발명의 효과들은 이상에서 언급한 효과들로 제한되지 않으며, 언급되지 않은 또 다른 효과들은 아래의 기재로부터 통상의 기술자에게 명확하게 이해 될 수 있을 것이다.

도면의 간단한 설명

[0027] 도 1은 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치가 포함하는 전체 구성을 예시적으로 도시한 도면이다.

도 2는 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법이 이루어지는 환경을 도시한 도면이다.

도 3은 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서, 데이터셋을 구축하는 S210 단계를 구체화한 순서도이다.

도 4는 OOXML 기반 문서 파일의 내부 구조를 도시한 도면이다.

도 5는 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서, 파일 이름을 수집하는 S210-3 단계를 구체화한 순서도이다.

도 6은 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서 수집한 파일 이름 각각에 대한 전처리를 수행하는 S210-4 단계를 구체화한 순서도이다.

도 7은 LSTM 네트워크 구조를 예시적으로 나타낸 도면이다.

도 8 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서 요구되는 학습 횟수에 대한 성능을 검증하기 위한 시뮬레이션 결과이다.

도 9는 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서 분류 결과의 정확도를 검증하기 위한 시뮬레이션 결과이다.

발명을 실시하기 위한 구체적인 내용

[0028] 본 발명의 목적과 기술적 구성 및 그에 따른 작용 효과에 관한 자세한 사항은 본 발명의 명세서에 첨부된 도면에 의거한 이하의 상세한 설명에 의해 보다 명확하게 이해될 것이다. 첨부된 도면을 참조하여 본 발명에 따른 실시 예를 상세하게 설명한다.

[0029] 본 명세서에서 개시되는 실시 예들은 본 발명의 범위를 한정하는 것으로 해석되거나 이용되지 않아야 할 것이다. 이 분야의 통상의 기술자에게 본 명세서의 실시 예를 포함한 설명은 다양한 응용을 갖는다는 것이 당연하다. 따라서, 본 발명의 상세한 설명에 기재된 임의의 실시 예들은 본 발명을 보다 잘 설명하기 위한 예시적인 것이며 본 발명의 범위가 실시 예들로 한정되는 것을 의도하지 않는다.

[0030] 도면에 표시되고 아래에 설명되는 기능 블록들은 가능한 구현의 예들일 뿐이다. 다른 구현들에서는 상세한 설명의 사상 및 범위를 벗어나지 않는 범위에서 다른 기능 블록들이 사용될 수 있다. 또한, 본 발명의 하나 이상의 기능 블록이 개별 블록들로 표시되지만, 본 발명의 기능 블록들 중 하나 이상은 동일 기능을 실행하는 다양한 하드웨어 및 소프트웨어 구성들의 조합일 수 있다.

[0031] 또한, 어떤 구성요소들을 포함한다는 표현은 "개방형"의 표현으로서 해당 구성요소들이 존재하는 것을 단순히 지칭할 뿐이며, 추가적인 구성요소들을 배제하는 것으로 이해되어서는 안 된다.

[0032] 나아가 어떤 구성요소가 다른 구성요소에 "연결되어" 있다거나 "접속되어" 있다고 언급될 때에는, 그 다른 구성요소에 직접적으로 연결 또는 접속되어 있을 수도 있지만, 중간에 다른 구성요소가 존재할 수도 있다고 이해되어야 한다.

- [0033] 이하에서는 도면들을 참조하여 본 발명의 세부적인 실시 예들에 대해 살펴보도록 한다.
- [0034] 도 1은 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100)가 포함하는 전체 구성을 예시적으로 도시한 도면이다.
- [0035] 그러나 이는 본 발명의 목적을 달성하기 위한 바람직한 실시 예일 뿐이며, 필요에 따라 일부 구성이 추가되거나 삭제될 수 있고, 어느 한 구성이 수행하는 역할을 다른 구성이 함께 수행할 수도 있음은 물론이다.
- [0036] 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100)는 프로세서(10), 네트워크 인터페이스(20), 메모리(30), 스토리지(40) 및 이들을 연결하는 데이터 버스(50)를 포함할 수 있으며, 기타 본 발명의 목적을 달성함에 있어 요구되는 부가적인 구성들을 더 포함할 수 있음은 물론이라 할 것이다.
- [0037] 프로세서(10)는 각 구성의 전반적인 동작을 제어한다. 프로세서(10)는 CPU(Central Processing Unit), MPU(Micro Processor Unit), MCU(Micro Controller Unit) 또는 본 발명이 속하는 기술 분야에서 널리 알려져 있는 형태의 프로세서 중 어느 하나일 수 있다. 아울러, 프로세서(10)는 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법을 수행하기 위한 적어도 하나의 애플리케이션 또는 프로그램에 대한 연산을 수행할 수 있다.
- [0038] 네트워크 인터페이스(20)는 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100)의 유무선 인터넷 통신을 지원하며, 그 밖의 공지의 통신 방식을 지원할 수도 있다. 따라서 네트워크 인터페이스(20)는 그에 따른 통신 모듈을 포함하여 구성될 수 있다.
- [0039] 메모리(30)는 각종 정보, 명령 및/또는 정보를 저장하며, 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법을 수행하기 위해 스토리지(40)로부터 하나 이상의 컴퓨터 프로그램(41)을 로드할 수 있다. 도 1에서는 메모리(30)의 하나로 RAM을 도시하였으나 이와 더불어 다양한 저장 매체를 메모리(30)로 이용할 수 있음은 물론이다.
- [0040] 스토리지(40)는 하나 이상의 컴퓨터 프로그램(41) 및 대용량 네트워크 정보(42)를 비임시적으로 저장할 수 있다. 이러한 스토리지(40)는 ROM(Read Only Memory), EPROM(Erasable Programmable ROM), EEPROM(Electrically Erasable Programmable ROM), 플래시 메모리 등과 같은 비휘발성 메모리, 하드 디스크, 착탈형 디스크, 또는 본 발명이 속하는 기술 분야에서 널리 알려져 있는 임의의 형태의 컴퓨터로 읽을 수 있는 기록 매체 중 어느 하나일 수 있다.
- [0041] 컴퓨터 프로그램(41)은 메모리(30)에 로드되어, 하나 이상의 프로세서(10)에 의해, (A) N(N은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 오퍼레이션, (B) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 오퍼레이션, (C) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 오퍼레이션, (D) 상기 수신한 마지막 저장 프로그램을 분류하고자 하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 오퍼레이션 및 (E) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 오퍼레이션을 실행하며, 상기 N개의 파일은, OOXML(Office Open XML) 기반 문서 파일일 수 있다.
- [0042] 이상 간단하게 언급한 컴퓨터 프로그램(41)이 수행하는 오퍼레이션은 컴퓨터 프로그램(41)의 일 기능으로 볼 수 있으며, 보다 자세한 설명은 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 대한 설명에서 후술하도록 한다.
- [0043] 데이터 버스(50)는 이상 설명한 프로세서(10), 네트워크 인터페이스(20), 메모리(30) 및 스토리지(40) 사이의 명령 및/또는 정보의 이동 경로가 된다.
- [0044] 이상 설명한 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100)는 독립된 디바이스의 형태, 예를 들어 전자 기기나 서버(클라우드 포함)의 형태일 수 있으며, 후자의 경우 전용 애플리케이션의 형태로 범용 수사를 담당하는 수사관의 전자 기기 등에 설치될 수 있을 것이다.
- [0045] 한편, 여기서 전자 기기는 스마트폰, 태블릿 PC, 노트북 PC, PDA, PMP 등과 같이 휴대가 용이한 포터블 기기 뿐만 아니라 한 장소에 고정 설치되어 사용하는 데스크톱 PC 등이라도 무방하며, 데이터의 수집 대상이 클라우드 기반 스토리지이기 때문에 네트워크 기능만 보유하고 있다면 전자 기기는 어떠한 것이라도 무방하다 할 것이다.
- [0046] 이하, 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100)가 독립된 디바이스 형태임을 전제로 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 대해

여 도 2 내지 도 7을 참조하여 설명하도록 한다.

- [0047] 도 2는 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법의 대표적인 단계를 나타낸 순서도이다.
- [0048] 그러나 이는 본 발명의 목적을 달성함에 있어서 바람직한 실시 예일 뿐이며, 필요에 따라 일부 단계가 추가 또는 삭제될 수 있음은 물론이고, 어느 한 단계가 다른 단계에 포함되어 수행될 수도 있다.
- [0049] 한편, 각 단계는 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100)를 통해 이루어지는 것을 전제로 하며, 설명의 편의를 위해 "장치(100)"로 명명하도록 함을 미리 밝혀두는 바이다.
- [0050] 우선, 장치(100)가 N개의 문서 파일을 포함하는 데이터셋을 구축한다(S210).
- [0051] 여기서 N은 2 이상의 정수인바, 그에 따라 데이터셋은 복수 개의 문서 파일을 포함하는 것으로 볼 수 있으며, N개의 문서 파일은 N개의 워드(Word) 문서 파일, N개의 파워포인트(Powerpoint) 문서 파일 및 N개의 엑셀(Excel) 문서 파일 중 어느 하나 이상일 수 있고, 기타 공지된 다른 종류의 문서 파일, 예를 들어 아래 한글 문서 파일 등을 더 포함할 수 있음은 물론이라 할 것이나, 워드 문서 파일, 파워포인트 문서 파일 및 엑셀 문서 파일이 문서 작업에 있어서 가장 보편적으로 널리 사용되고 있는 문서 파일이기 때문에 데이터셋 구축에 가장 중요한 역할을 수행할 수 있다 할 것이다.
- [0052] 한편, N개의 문서 파일이 N개의 워드 문서 파일, N개의 파워포인트 문서 파일 및 N개의 엑셀 문서 파일 중 어느 하나 이상이면 데이터셋을 추구함에 있어 충분하기는 하나, 데이터셋은 후술할 S220 단계에서 분류 모델이 입력되어 학습에 활용되기 때문에 학습 효율을 향상시키기 위해 최대한 다양한 종류의 문서 파일을 포함하는 것이 바람직할 것인바, 어느 한 종류의 문서 파일만을 포함하는 것보다는 워드 문서 파일, 파워포인트 문서 파일 및 엑셀 문서 파일을 전부 포함하는 것이 가장 바람직하다 할 것이다.
- [0053] 이하, 데이터셋의 구축 과정에 대하여 보다 자세히 설명하도록 한다.
- [0054] 도 3은 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서, 데이터셋을 구축하는 S210 단계를 구체화한 순서도이다.
- [0055] 그러나 이는 본 발명의 목적을 달성함에 있어서 바람직한 실시 예일 뿐이며, 필요에 따라 일부 단계가 추가 또는 삭제될 수 있음은 물론이고, 어느 한 단계가 다른 단계에 포함되어 수행될 수도 있다.
- [0056] 우선, 장치(100)가 N개의 문서 파일을 수집한다(S210-1).
- [0057] 여기서 N개의 문서 파일 수집은 장치(100)의 관리자가 자체적으로 보유한 파일들로부터 수집할 수 있으며 또는 온라인에 배포된 문서 파일들을 크롤링(Crawling)을 통해 수집할 수 있음은 물론인바, 수집 경로는 어떠한 경우라도 무방하다 할 것이다.
- [0058] 이하의 설명에서는 장치(100)가 500개의 워드 문서 파일, 500개의 파워포인트 문서 파일 및 500개의 엑셀 문서 파일을 수집한 경우를 예로 하여 설명을 이어가도록 한다.
- [0059] 이후, 수집한 N개의 문서 파일 각각을 장치(100)가 M개의 문서 생성/수정 프로그램 별로 수정하여 저장한다(S210-2).
- [0060] 여기서 M 역시 2 이상의 정수인바, 워드 문서 파일을 생성하거나 수정할 수 있는 공지된 모든 프로그램, 파워포인트 문서 파일을 생성하거나 수정할 수 있는 공지된 모든 프로그램, 엑셀 문서 파일을 생성하거나 수정할 수 있는 공지된 모든 프로그램을 전부 포함하는 것이 바람직하며, 응용프로그램의 종류에 따라 워드 문서 파일만을 생성하거나 수정할 수 있는 프로그램, 파워포인트 파일만을 생성하거나 수정할 수 있는 프로그램, 엑셀 문서 파일만을 생성하거나 수정할 수 있는 프로그램이 존재할 수 있으므로 M은 문서 파일의 종류에 따라 그 값이 각각 상이해질 수 있다 할 것이다.
- [0061] 예를 들어, 워드 문서 파일에 대해서는 문서 생성/수정 프로그램이 마이크로소프트 오피스, 마이크로소프트 오피스(Mac), 리브레 오피스, 리브레 오피스(Mac), 폴라리스 오피스, 구글 워크스페이스, 한컴 오피스, 네이버 오피스, 오피스수트, 프리 오피스, WPS 오피스, 온리 오피스, 워드패드, 스마트워드, 텍스트에디터(Mac)일 수 있으며, 이 경우 M은 15임에 반해, 파워포인트 문서 파일 및 엑셀 문서 파일은 이들 문서 생성/수정 프로그램 중, 텍스트에디터(Mac)를 통해 문서 파일을 생성하거나 수정할 수는 없으므로 파워포인트 문서 파일 및 엑셀 문서 파일에 대해서는 M이 12(마이크로소프트 오피스, 마이크로소프트 오피스(Mac), 리브레 오피스, 리브레 오피스

(Mac), 폴라리스 오피스, 구글 워크스페이스, 한컴 오피스, 네이버 오피스, 오피스수트, 프리 오피스, WPS 오피스, 온리 오피스)일 수 있을 것이며, 이하 이를 예로 하여 설명을 이어가도록 한다.

[0062] 한편, 앞서 S210-1 단계에서 500개의 워드 문서 파일, 500개의 파워포인트 문서 파일, 500개의 엑셀 문서 파일을 예시적으로 수집했다고 한바, 500개의 워드 문서 파일 각각에 대하여 마이크로소프트 오피스, 마이크로소프트 오피스(Mac), 리브레 오피스, 리브레 오피스(Mac), 폴라리스 오피스, 구글 워크스페이스, 한컴 오피스, 네이버 오피스, 오피스수트, 프리 오피스, WPS 오피스, 온리 오피스, 워드패드, 스마트워드, 텍스트에디터(Mac) 별로 수정하여 저장하고(전체 $500 * 15 = 7500$ 회 수정하여 저장), 500개의 파워포인트 문서 파일에 대하여 마이크로소프트 오피스, 마이크로소프트 오피스(Mac), 리브레 오피스, 리브레 오피스(Mac), 폴라리스 오피스, 구글 워크스페이스, 한컴 오피스, 네이버 오피스, 오피스수트, 프리 오피스, WPS 오피스, 온리 오피스 별로 수정하여 저장하며(전체 $500 * 12 = 6000$ 회 수정하여 저장), 500개의 엑셀 문서 파일에 대하여 마이크로소프트 오피스, 마이크로소프트 오피스(Mac), 리브레 오피스, 리브레 오피스(Mac), 폴라리스 오피스, 구글 워크스페이스, 한컴 오피스, 네이버 오피스, 오피스수트, 프리 오피스, WPS 오피스, 온리 오피스 별로 수정하여 저장(전체 $500 * 12 = 6000$ 회 수정하여 저장)하는 것이다.

[0063] 이와 같이 N개의 문서 파일을 M개의 문서 생성/수정 프로그램 별로 수정하여 저장하는 것은 OOXML 기반 문서 파일(200)의 내부 구조를 활용하기 위함인바, 이하 설명하도록 한다.

[0064] 도 4는 OOXML 기반 문서 파일(200)의 내부 구조를 도시한 도면이다.

[0065] 장치(100)가 수집하는 N개의 문서 파일은 OOXML 기반 문서 파일이며, OOXML 포맷은 PKWARE사의 PKZIP 포맷을 기반으로 만들어졌기 때문에 OOXML 포맷은 PKZIP 포맷과 동일한 구조를 지니며, 여기서 PKZIP 포맷은 압축 파일에서 활용하는 대표적인 압축 파일 포맷이다.

[0066] 도 4를 참조하면, OOXML 기반 문서 파일은 로컬 파일 헤더(Local File Header, 210), 중앙 디렉터리 파일 헤더(Central Directory File Header, 220) 및 마지막 중앙 디렉터리 레코드(End of Central Directory File Header, 230)로 이루어져 있음을 확인할 수 있다.

[0067] 여기서 로컬 파일 헤더는 압축된 파일들의 이름, 마지막 수정 시간, 압축 크기, 압축 방식 등과 같은 압축 파일들에 대한 다양한 정보들이 저장되며, 중앙 디렉터리 파일 헤더는 로컬 파일 헤더와 같이 압축된 파일들의 이름, 마지막 수정 시간, 압축 크기, 압축 방식뿐만 아니라 기타 속성 정보나 로컬 파일 헤더의 위치 정보 등이 추가적으로 저장될 수 있고, 마지막 중앙 디렉터리 레코드는 중앙 디렉터리 파일 헤더(Central Directory Header)의 시작 위치나 개수 등이 저장될 수 있다.

[0068] OOXML 기반 문서 파일은 압축 파일 포맷을 기반으로 만들어졌기 때문에 여러 파일들이 압축된 형태를 지니고 있는바, 그에 따라 해당 파일 내부에는 문서 파일을 구성하는 다양한 파일들이 존재하며, 해당 파일들에 대한 정보는 로컬 파일 헤더에 저장된다.

[0069] 보다 구체적으로, OOXML 문서 파일을 문서 생성/수정 프로그램을 통해 수정하여 저장하면 새로운 파일을 생성한 뒤 수정한 문서 내용을 덮어쓰기 파일 이름과 함께 저장하는데 문서 생성/수정 프로그램마다 문서 파일 내 로컬 파일 헤더에 저장하는 파일들의 순서가 전부 상이하다는 특징을 가지고 있으며, 본 발명에서는 이와 같은 OOXML 기반 문서 파일의 특징을 활용하여 마지막 저장 프로그램을 분류하기 위해 N개의 문서 파일 각각을 M개의 문서 생성/수정 프로그램 별로 수정하여 저장하는 것이다.

[0070] 다시 도 3에 대한 설명으로 돌아가도록 한다.

[0071] N개의 문서 파일 각각을 M개의 문서 생성/수정 프로그램 별로 수정하여 저장했다면, 장치(100)가 M개의 문서 생성/수정 프로그램 별로 수정하여 저장한 N개의 파일 각각의 로컬 파일 헤더 내 파일 이름(File Name)을 M개의 문서 생성/수정 프로그램 별로 순서대로 수집한다(S210-3).

[0072] 이와 같은 S210-3 단계는 도 5에 도시된 바와 같이 M개의 문서 생성/수정 프로그램 별로 수정하여 저장한 N개의 파일 각각의 로컬 파일 헤더 내 파일 이름 길이(File Name Length)를 순서대로 확인하는 단계(S210-3-1) 및 파일 이름 길이를 확인한 N개의 파일 각각의 로컬 파일 헤더 내 파일 이름을 M개의 문서 생성/수정 프로그램 별로 순서대로 수집하는 단계(S210-3-2)를 포함하는바, S210-3-1 단계에서 파일 이름 길이를 먼저 확인하는 이유는 파일 이름 자체가 파일마다 가변적이기 때문에 파일 이름 길이를 먼저 확인하고 그에 해당하는 만큼의 파일 이름만 읽어내기 위함이다. 예를 들어, 확인한 파일 이름 길이가 10이라면, 파일 이름도 길이가 10에 해당하는 부분까지만 읽어내면 된다는 것이다.

- [0073] 한편, S210-3 단계를 앞선 예에 대입하면, 500개의 워드 문서 파일에 대하여 15개의 문서 생성/수정 프로그램 별로 수정하여 저장했기에 7500개의 파일 이름을 순서대로 수집할 수 있을 것이며, 500개의 파워포인트 문서 파일에 대하여 12개의 문서 생성/수정 프로그램 별로 수정하여 저장했기에 6000개의 파일 이름을 순서대로 수집할 수 있을 것이고, 500개의 엑셀 문서 파일에 대하여 12개의 문서 생성/수정 프로그램 별로 수정하여 저장했기에 6000개의 파일 이름을 순서대로 수집할 수 있을 것이다.
- [0074] 파일 이름까지 수집했다면, 장치(100)가 M개의 문서 생성/수정 프로그램 별로 수집한 파일 이름 각각에 대한 전처리를 수행한다(S210-4).
- [0075] 여기서 전처리는 후술할 분류 모델에 입력하기 전에 분류 모델이 이해할 수 있는 형태 또는 일정한 형태로 데이터셋을 가공해주는 작업을 의미하는바, 이하 도 6을 참조하여 설명하도록 한다.
- [0076] 도 6은 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서 수집한 파일 이름 각각에 대한 전처리를 수행하는 S210-4 단계를 구체화한 순서도이다.
- [0077] 그러나 이는 본 발명의 목적을 달성함에 있어서 바람직한 실시 예일 뿐이며, 필요에 따라 일부 단계가 추가 또는 삭제될 수 있음은 물론이고, 어느 한 단계가 다른 단계에 포함되어 수행될 수도 있다.
- [0078] 도 6을 참조하면, S210-4 단계는 수집한 파일 이름 각각이 동일한 파일 이름과 서로 다른 숫자들이 조합된 경우, 가장 처음 확인된 파일 이름만 숫자를 제거하고, 그 이후에 확인된 나머지 파일 이름은 전부 삭제하는 전처리를 수행하는 단계(S210-4-1)를 포함할 수 있다.
- [0079] 예를 들어, 수집한 파일이 사진을 다수 포함하는 문서 파일인 경우, 파일 이름이 image1.jpeg, image2.jpeg, image3.jpeg 꺾 imageN.jpeg 등과 같이 저장될 수 있는데 이러한 경우 문서 파일의 이름 순서 및 특성으로 인해 마지막 저장 프로그램 분류 시, 그 결과에 영향을 주기 때문에 가장 처음 확인된 파일 이름만 숫자를 제거하고, 그 이후에 확인된 나머지 파일 이름은 전부 삭제할 수 있는바, 이상의 예를 기준으로 설명하면, image1.jpeg에서 1을 제거한 image.jpeg만을 저장하고 image2.jpeg, image3.jpeg 꺾 imageN.jpeg는 전부 삭제하는 것이다.
- [0080] 또한, S210-4 단계는 수집한 파일 이름 각각이 대문자 및 소문자를 혼용하여 조합된 경우, 파일 이름 각각을 전부 소문자로 통일시키는 전처리를 수행하는 단계(S210-4-2)를 포함할 수 있다.
- [0081] 문서 생성/수정 프로그램 별로 각자의 파일 이름 명명규칙을 보유하고 있는바, 특정 문서 파일에 대하여 어떠한 문서 생성/수정 프로그램으로 문서 파일을 마지막으로 저장했는지 분류하기 위해서는 파일 이름 명명규칙을 통일시켜야할 필요성이 있는바, 이를 위해 대문자 및 소문자를 혼용하여 조합된 파일 이름을 소문자로 통일시키는 것이며, 이와 반대로 대문자로 통일시킬 수도 있음은 물론이라 할 것이다.
- [0082] 더 나아가, S210-4 단계는 수집한 파일 이름 각각이 불용어를 포함하여 조합된 경우, 각각의 파일 이름에서 불용어를 삭제하는 전처리를 수행하는 단계(S210-4-3)를 포함할 수 있다.
- [0083] 불용어란 문장의 구성에서 큰 의미를 갖지 아니하되, 파일 이름에 사용할 수 있는 단어나 기호를 의미하는바, 예를 들어 [,], ' 등과 같은 기호를 들수 있으며, 이뿐만 아니라 문서 생성/수정 프로그램 별로 사용되는 다양한 불용어들을 삭제할 수도 있다 할 것이다.
- [0084] 이상 설명한 바와 같은 전처리를 수행했다면, 장치(100)가 M개의 문서 생성/수정 프로그램 별로 파일 이름이 전처리된 N개의 문서 파일이 대응된 데이터셋을 구축한다(S210-5).
- [0085] 이상 설명한 데이터셋의 구축은 S210-1 단계 내지 S219-5 단계를 통해 구축하는 것이 후술할 학습의 측면에서 바람직할 수 있으나, 이와 별개로 외부 장치(미도시) 등을 통해 기 구축한 데이터셋을 로드해올 수도 있다 할 것이다.
- [0086] 다시 도 2에 대한 설명으로 돌아가도록 한다.
- [0087] 데이터셋을 구축했다면, 장치(100)가 구축한 데이터셋을 분류 모델에 입력하여 학습한다(S220).
- [0088] 여기서 분류 모델은 장단기 메모리인 LSTM(Long Short Term Memory) 네트워크 구조를 갖는 딥러닝 기반 분류 모델일 수 있으며, 도 7에 LSTM 네트워크 구조를 예시적으로 도시해 놓았다.
- [0089] 도 7을 참조하여 LSTM 네트워크 구조에 대하여 설명하면, 데이터셋이 포함하는 전처리한 파일 이름들이 x^t 에 입력되며, σ 와 h 는 LSTM 네트워크의 시그모이드와 하이퍼 볼릭 탄젠트가 활성화 함수로 숨겨진 은닉층을 의미한

다. 아울러, c^t 는 이전 상태를 장기적으로 기억하기 위한 데이터이며 o^t 는 연산에 의한 결과이다. 이와 같이 LSTM 네트워크는 현재 입력과 이전 출력을 동시에 받아들이기 때문에 시계열 데이터 분석에 용이한바, 앞서 S210-1-3 단계에서 로컬 파일 헤더 내 파일 이름을 순서대로 수집하였으며, 이는 시계열 데이터에 해당하기 때문에 LSTM 네트워크 구조가 분류 모델에 최적화된 네트워크 구조로 볼 수 있는 것이다.

[0090] 한편, 분류 모델의 학습은 LSTM 네트워크가 데이터셋이 포함하는 전처리된 파일 이름들의 순서 정보를 참조하여 이루어진다. 분류 모델에 학습하기 위해 텍스트 기반 데이터인 전처리된 파일 이름들을 원-핫 인코딩(One-Hot Encoding) 방식을 활용하여 학습 모델이 쉽게 식별할 수 있는 데이터로 변환해줘야 하며, 여기서 원-핫 인코딩이란 컴퓨터 또는 기계가 자연어를 처리할 때 연산하기 용이하도록 문자를 숫자로 변환해주는 것을 의미한다. 이렇게 숫자 데이터로 변환한 파일 이름들은 여전히 시계열 데이터이기 때문에 LSTM 네트워크에 학습시킬 수 있으며, 학습된 LSTM 네트워크를 분류층에서 활용하여 문서가 어떠한 문서 생성/수정 프로그램을 통해 마지막으로 저장되었는지 분류할 수 있다. 이 경우, 분류 모델의 최적의 하이퍼파라미터(Hyperparameter)는 LSTM Cell이 30, Dropout rate는 0.2, Activation Function은 Softmax, Optimizer는 Adam, Loss Function은 Categorical Crossentropy, Epoch는 10 및 Batch Size는 64일 수 있으나, 이는 예시적인 것에 해당한다 할 것이다.

[0091] 분류 모델의 학습이 완료되었다면 장치(100)가 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하며 (S230), 수신한 마지막 저장 프로그램을 분류하고자 하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고 이에 대한 전처리를 수행하고(S240), 전처리를 수행한 마지막 저장 프로그램을 분류하고자하는 문서 파일을 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력한다(S250).

[0092] 이 경우, S230 단계에서 수신하는 문서 파일 역시 OOXML 기반의 문서 파일, 보다 구체적으로 워드 문서 파일, 파워포인트 문서 파일 및 엑셀 문서 파일 중 어느 하나일 것이며, S240 단계에서 수행하는 파일 이름의 수집은 S210-3 단계에서, 전처리하는 S210-1-4 단계에서 설명한 것과 동일하므로 중복 서술을 방지하기 위해 자세한 설명은 생략하도록 한다.

[0093] 아울러, S250 단계 이후에, 마지막 저장 프로그램 분류 결과를 재학습하는 단계(미도시)가 수행될 수 있으며, 그에 따라 장치(100)는 마지막 저장 프로그램을 분류하기 위해 사용하면 할수록 분류 결과의 정확도가 향상될 수 있을 것이다.

[0094] 지금까지 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 대하여 설명하였다. 본 발명에 따르면, 데이터셋을 구축하여 분류 모델에 학습시킨 이후, 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하여 분류 모델에 입력하기만 하면 분류 결과가 손쉽게 출력되는바, 과제는 디지털포렌식 활동에 있어서 문서 파일의 근원지를 파악할 때 용이하게 활용할 수 있다. 또한, 비교적 적은 개수의 문서 파일을 통해 분류 모델의 학습이 가능하기에 학습에 필요한 데이터의 크기가 크지 않고, 문서 파일의 마지막 저장 프로그램 분류시 활용하는 데이터의 양까지 적게할 수 있는바, 수사관이 현장에서 활용할 수 있는 이동형 디지털포렌식 분석 시스템에 손쉽게 탑재함으로써 편의성을 향상시킬 수 있다. 아울러, 분류 모델의 학습이 신속하게 이루어질 수 있으며, 사용하면 할수록 문서 파일의 마지막 저장 프로그램 분류 결과를 지속적으로 학습하기 때문에 분류의 정확도가 향상될 수 있음과 동시에 분류 결과의 출력 역시 신속하게 이루어질 수 있다

[0095] 는 효과가 있다.

[0096] 도 8 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서 요구되는 학습 횟수에 대한 성능을 검증하기 위한 시뮬레이션 결과인바, 도 8을 참조하면 워드 문서 파일, 엑셀 문서 파일 및 파워포인트 문서 파일 모두 8회 이하의 학습만을 수행하더라도 분류 결과의 정확도(accuracy)가 우수한 상태에 도달함을 확인할 수 있으며(1에 수렴), 이와 마찬가지로 워드 문서 파일, 엑셀 문서 파일 및 파워포인트 문서 파일 모두 8회 이하의 학습만을 수행하더라도 분류 결과의 오류(loss) 역시 비약적으로 감소함을 확인할 수 있다(0에 수렴).

[0097] 한편, 도 8을 보다 구체적으로 살펴보면 워드 문서 파일은 4회 정도의 학습을 통해, 엑셀 문서 파일은 6회 정도의 학습을 통해, 파워포인트 문서 파일은 8회 정도의 학습을 통해 분류 결과의 정확도가 우수한 상태에 도달함을 확인할 수 있는바(train_acc가 val_acc에 도달함), 이는 워드 문서 파일의 경우 다른 문서 파일들보다 구별 가능한 특징이 명확하게 존재한다는 것을 의미하나, 엑셀 문서 파일과 파워포인트 문서 파일 역시 학습 횟수에 비추어 보았을 때 보편적인 인공지능 모델의 학습에 요구되는 횟수보다 현저하게 적기 때문에 전체적인 발명의 성능이 우수함을 확인할 수 있다 할 것이다.

[0098] 아울러, 도 8에 도시된 시뮬레이션을 수행한 환경을 설명하면 운영체제는 Windows 10 Education, CPU는

Intel(R) Core(TM) i5-7500 CPU @ 3.50 GHz, RAM은 16GB (DDR4)이며, 500개의 워드 문서 파일의 데이터셋 크기는 3.99MB, 학습 속도는 23.1초, 500개의 엑셀 문서 파일의 데이터셋 크기는 2.25MB 이며 학습 속도는 19.6 초, 500개의 파워포인트 문서 파일의 데이터셋 크기는 4.59MB 이며 학습 속도는 16.9 초였던바, 매우 적은 학습 데이터셋만으로도 신속하게 학습이 이루어질 수 있음을 확인할 수 있었다. 또한, 워드 문서 파일 500개의 분류 결과를 출력하는데 0.2초, 엑셀 문서 파일 500개의 분류 결과를 출력하는데 0.5초, 파워포인트 문서 파일 500개의 분류 결과를 출력하는데 4.2초가 소요된바, 분류 결과 역시 신속하게 이루어짐을 확인할 수 있었다.

[0099] 도 9는 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법에 있어서 분류 결과의 정확도를 검증하기 위한 시뮬레이션 결과인바, 1,500개의 워드 문서 파일의 경우 마이크로소프트 오피스 프로그램인데 마이크로소프트 오피스(Mac) 프로그램으로 분류한 4개의 워드 문서 파일, 마이크로소프트 오피스(Mac) 프로그램인데 마이크로소프트 오피스 프로그램으로 분류한 5개의 문서 파일, 오피스 수트 프로그램인데 마이크로소프트 오피스 프로그램으로 분류한 1개의 워드 문서 파일을 제외하고 정확하게 마지막 저장 프로그램 분류 결과를 출력했음을 확인할 수 있으며, 1,500개의 엑셀 문서 파일의 경우 마이크로소프트 오피스(Mac) 프로그램인데 마이크로소프트 오피스 프로그램으로 분류한 1개의 엑셀 문서 파일, 리브레 오피스(Mac) 프로그램인데 리브레 오피스 프로그램으로 분류한 3개의 엑셀 문서 파일, 오피스 수트 프로그램인데 구글 워크스페이스로 분류한 3개의 엑셀 문서 파일을 제외하고 정확하게 마지막 저장 프로그램 분류 결과를 출력했음을 확인할 수 있고, 1,500개의 파워포인트 문서 파일의 경우 마이크로소프트 프로그램인데 마이크로소프트(Mac) 프로그램으로 분류한 1개의 파워포인트 문서 파일, 마이크로소프트(Mac) 프로그램인데 마이크로소프트 프로그램으로 분류한 3개의 파워포인트 문서 파일, 리브레 오피스(Mac) 프로그램인데 리브레 오피스 프로그램으로 분류한 1개의 파워포인트 문서 파일을 제외하고 정확하게 마지막 저장 프로그램 분류 결과를 출력했음을 확인할 수 있는바, 분류의 정확도는 99% 이상으로써 매우 높음을 확인할 수 있다.

[0100] 한편, 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100)와 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법은 모든 기술적 특징을 동일하게 포함하는 본 발명의 제3 실시 예에 따른 컴퓨터로 판독 가능한 매체에 저장된 컴퓨터 프로그램으로 구현할 수도 있는바, 이 경우 컴퓨팅 장치와 결합하여, (AA) N(N은 2 이상의 정수)개의 문서 파일을 포함하는 데이터셋을 구축하는 단계, (BB) 상기 구축한 데이터셋을 분류 모델에 입력하여 학습하는 단계, (CC) 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 수신하는 단계, (DD) 상기 수신한 마지막 저장 프로그램을 분류하고자하는 문서 파일의 파일 이름을 로컬 파일 헤더에서 수집하고, 이에 대한 전처리를 수행하는 단계 및 (EE) 상기 전처리를 수행한 마지막 저장 프로그램을 분류하고자 하는 문서 파일을 상기 분류 모델에 입력하여 마지막 저장 프로그램 분류 결과를 출력하는 단계를 실행할 수 있을 것이며, 중복 서술을 위해 자세히 기재하지는 않았지만 본 발명의 제1 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 장치(100) 및 본 발명의 제2 실시 예에 따른 문서 파일의 마지막 저장 프로그램을 분류하는 방법 에 적용된 모든 기술적 특징은 본 발명의 제3 실시 예에 따른 컴퓨터로 판독 가능한 매체에 저장된 컴퓨터 프로그램에 모두 동일하게 적용될 수 있음은 물론이라 할 것이다.

[0101] 이상 첨부된 도면을 참조하여 본 발명의 실시 예들을 설명하였지만, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자는 본 발명이 그 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 실시될 수 있다는 것을 이해할 수 있을 것이다. 그러므로 이상에서 기술한 실시 예들은 모든 면에서 예시적인 것이며 한정적이 아닌 것으로 이해해야만 한다.

부호의 설명

[0102] 10: 프로세서
20: 네트워크 인터페이스
30: 메모리
40: 스토리지
41: 컴퓨터 프로그램
50: 정보 버스
100: 문서 파일의 마지막 저장 프로그램을 분류하는 장치
200: OOXML 기반 문서 파일

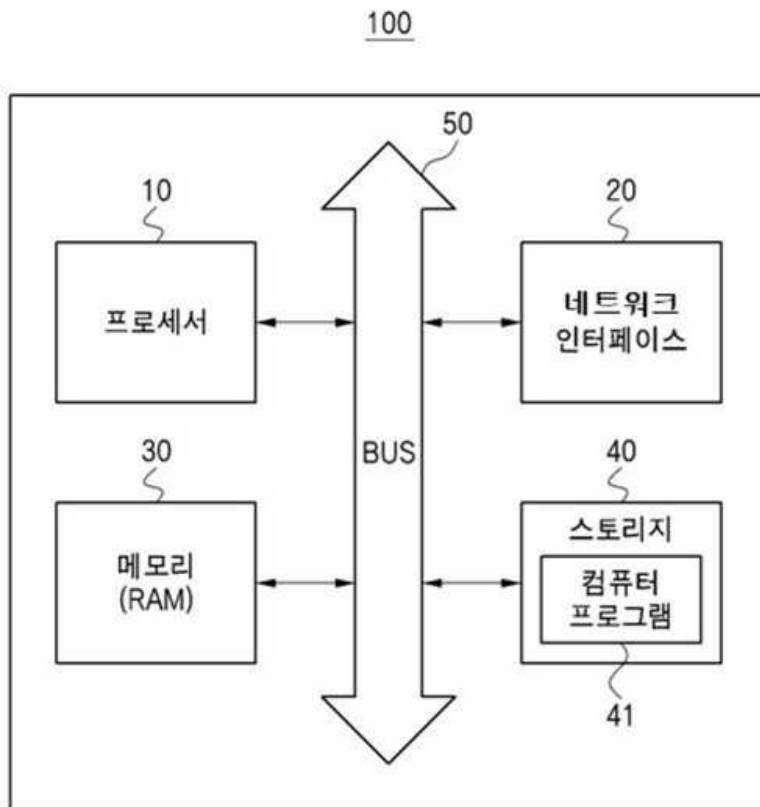
210: 로컬 파일 헤더

220: 중앙 디렉터리 파일 헤더

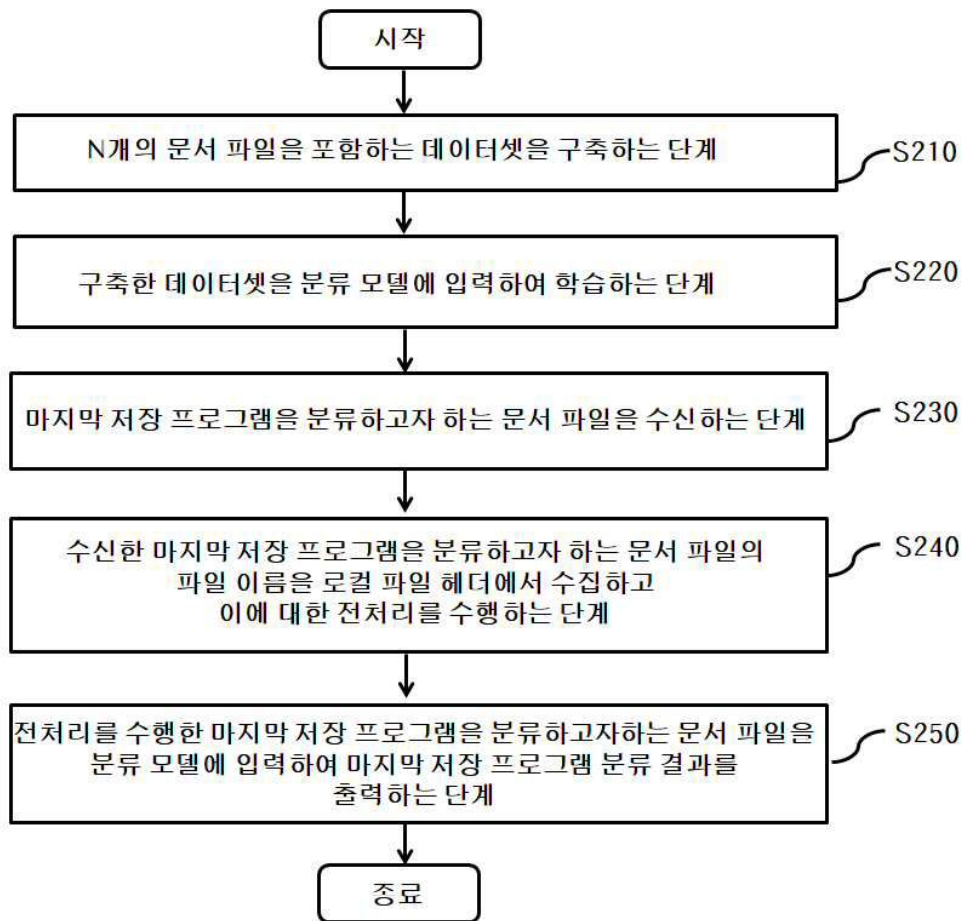
230: 마지막 중앙 디렉터리 레코드

도면

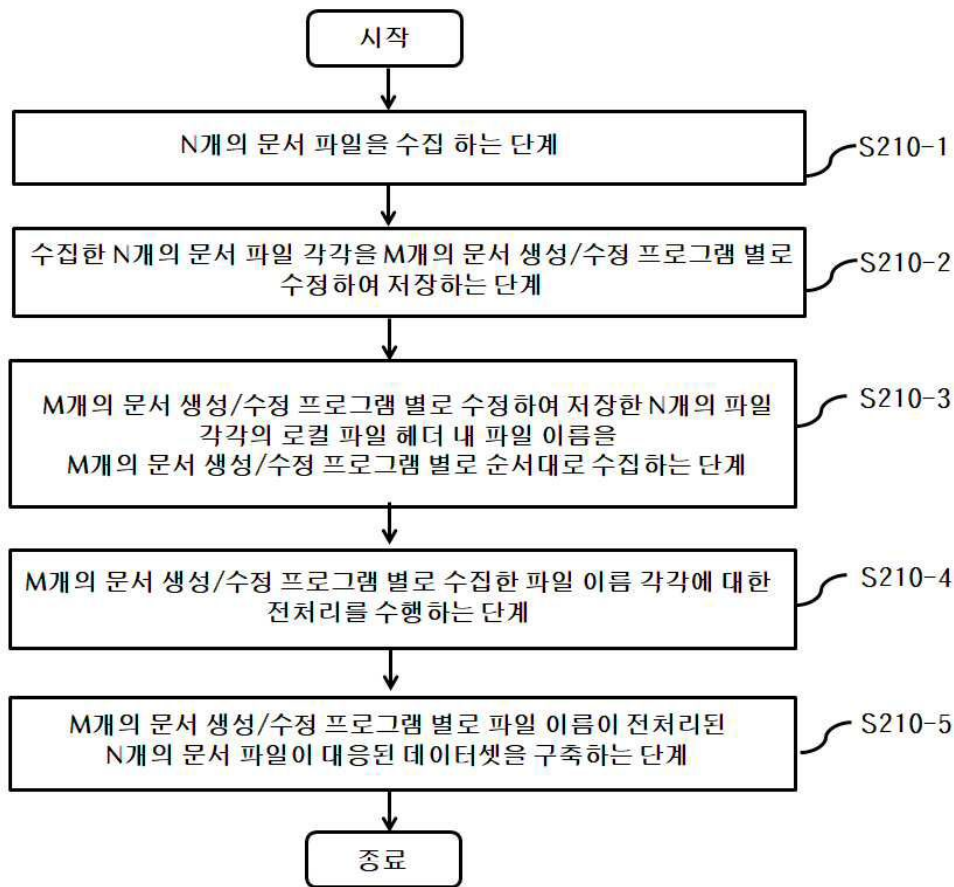
도면1



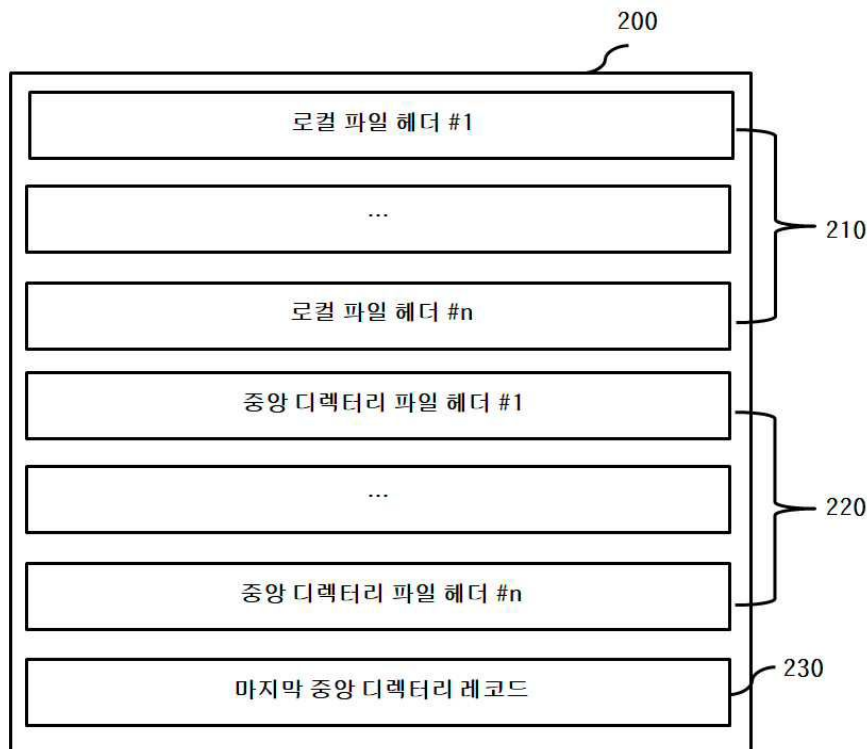
도면2



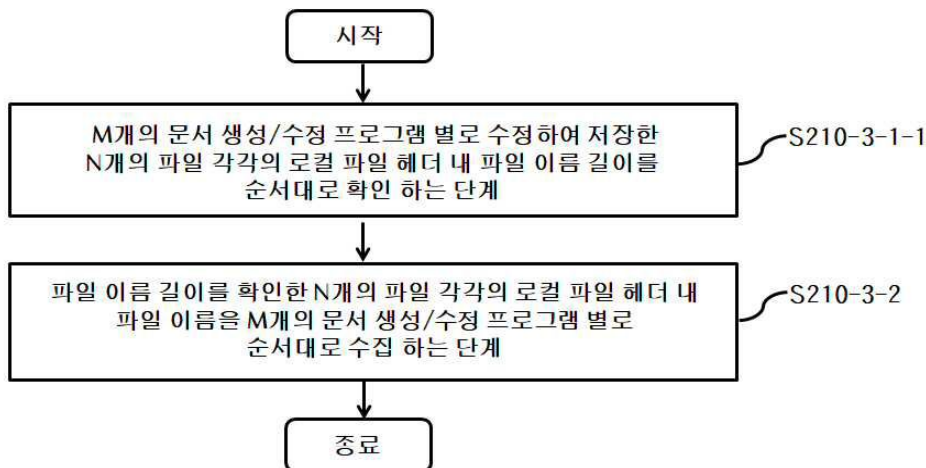
도면3



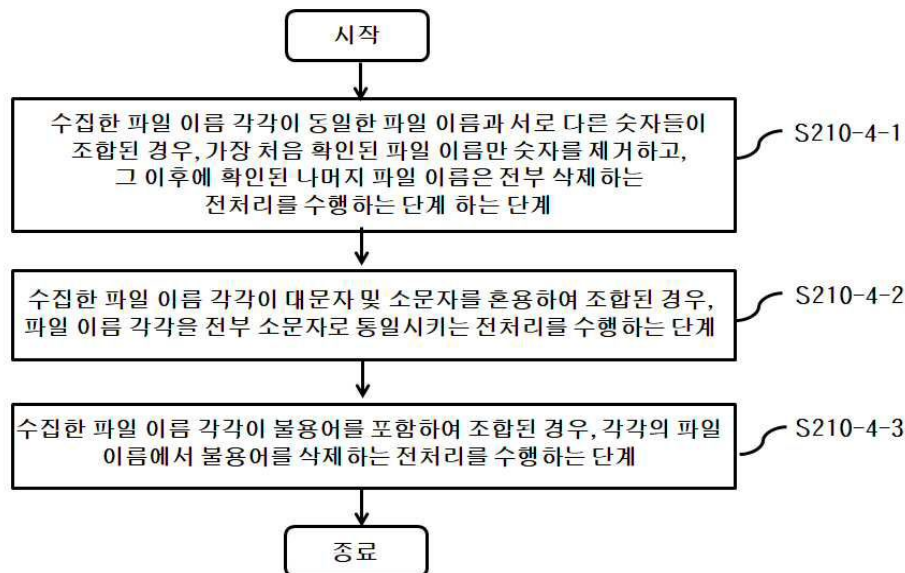
도면4



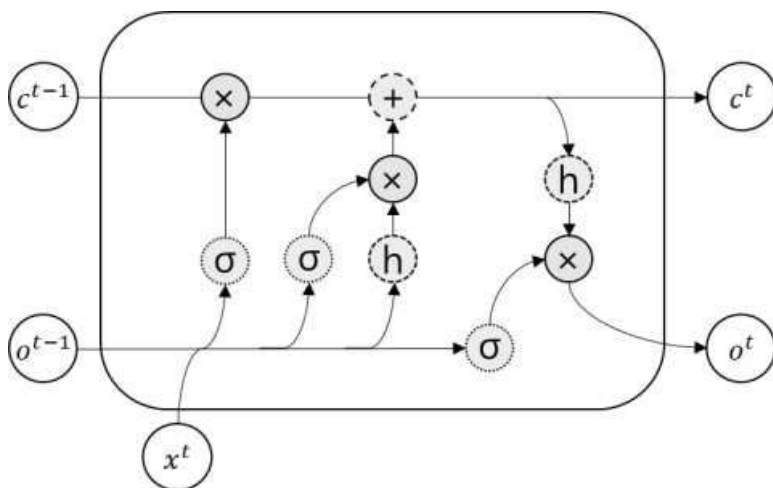
도면5



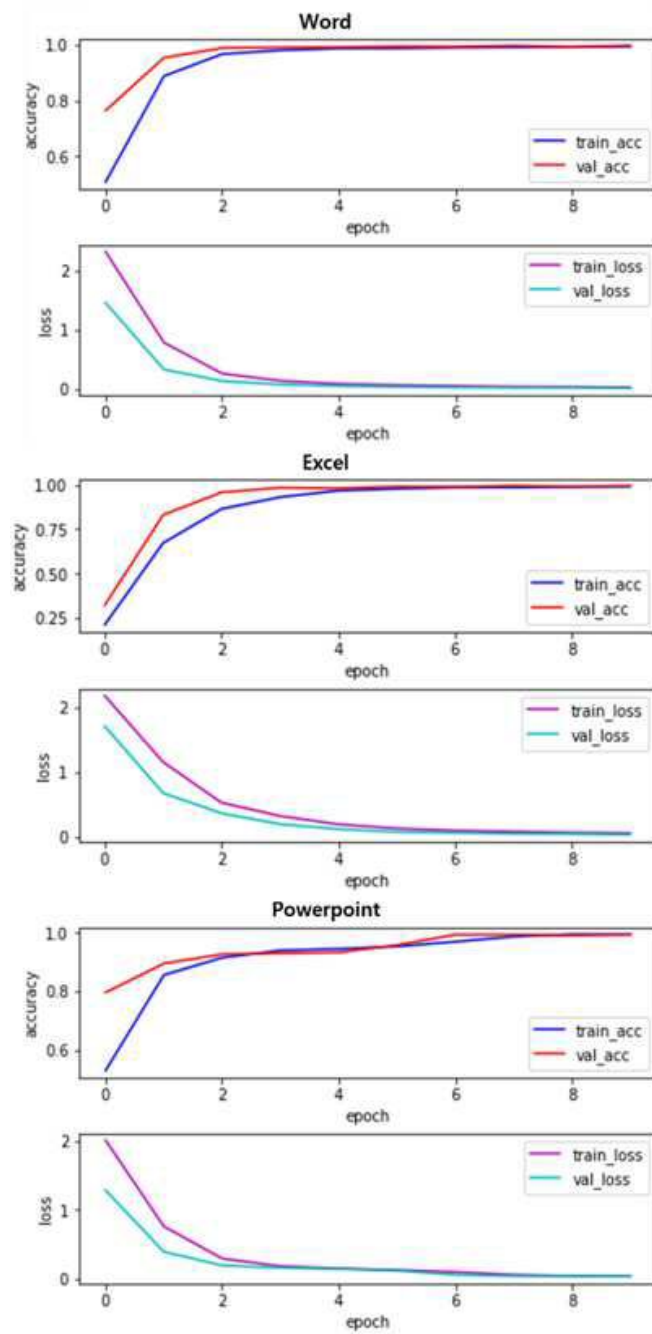
도면6



도면7



도면8



도면9

