Smartthings(IOT platform – Server Application) 화재 위험 관리: 서버 사이드 애플리케이션의 CI/CD*

김승주[1], 김지현[1], 박지원[2], 신현정[1], 김윤희[1]
[1]숙명여자대학교 소프트웨어학부, [2]숙명여자대학교 통계학과
e-mail : 4936joo@naver.com, were1117@sookmyung.ac.kr, orilove1@sookmyung.ac.kr

Smartthings (IOT platform – Server Application) Fire Risk Management: CI/CD of Server-side Applications

Seunju Kim[1], Jihyun Kim[1], Jiwon Park[2], Hyunjeong Shin[1], Yoonhee Kim[1]

1. 서 론

현대 정보 기술의 발전과 함께 스마트 디바이스와 클라우드 컴퓨팅은 우리의 일상 생활과 산업 전반에 혁신을 가져오고 있다. 특히, 스마트 홈 시스템과 같은 분야에서는 데이터의 실시간 분석과 효과적인 배포가 중요한 역할을 한다. 이러한 배경에서 스마트 디바이스와 클라우드 네이티브 기술을 결합하여 보다 효율적인 시스템을 구축하는 것이 필요하다. 본 논문에서는 SmartThings와 Kubernetes를 활용한 CI/CD 접근 방식을 통해 시스템의 효율적인 개발, 업데이트 및 배포를 지원하는 방법을 제안한다.

Kubernetes 기반의 CI/CD 환경에서 무중단 자동화 배포 전략[1]에서는 애플리케이션의 자동화 배포 환경을 개선하고, 다양한 배포 전략 실험을 통해 무중단 배포 전략 수립의 가능성을 확인할 수 있다. 또한, 스마트폰 앱 SmartThings는 SmartThings Energy 서비스를 통해 전력 데이터와 가전 데이터의 통합 관리를 추진하며, 신서비스 개발을 확대하고자 한다.[2] 본 연구에서는 Kubernetes를 활용하여 분산된 전기 자원의 상태를 실시간으로 모니터링하고, 서버 애플리케이션에 이미지를 배포하여 관리하는 방안을 모색한다. 이를 통해 스마트홈 환경에서의 데이터 학습 및 통합 관리를 가능하게 하고자 한다.

Docker는 컨테이너화 기술로 소프트웨어 개발 및 배포 과정에서 사용되고 있다. Docker는 애플리케이션과 그 종속성을 패키징하여 일관된 환경에서 실행할 수 있게 해준다. Docker는 마이크로 서비스 아키텍처, 클라우드 네이티브 애플리케이션 개발, 그리고 DevOps 실무에서 핵심적인 역할을 한다.

Kubeflow는 Kubernetes 위에서 머신러닝 워크플로우를 구축하고 배포하기 위한 오픈소스 플랫폼이다. 머신러닝 프로젝트의 확장성과 재현성을 위해 이를 채택한다. Kubeflow는 데이터 준비, 모델 학습, 하이퍼파라미터 튜닝, 모델 서빙 등 머신러닝 라이프 사이클의 전 과정을 관리할 수 있게 해주어, 데이터 과학자들의 생산성을 크게 향상시키고 있다.

GitHub Actions는 소프트웨어 개발 워크플로우를 자동화하기 위한 CI/CD 플랫폼이다. GitHub Actions의 주요 장점은 GitHub 저장소와의 긴밀한 통합, 다양한 언어 및 플랫폼 지원, 그리고 사용자 정의 가능한 워크플로우에 해당한다. 개발자들은 코드 테스트, 빌드, 배포 등의 과정을 자동화하여 개발 생산성의 향상을 도모한다.

SmartThings는 삼성 전자의 IoT 플랫폼으로, 스마트홈 시장에서 중요한 위치를 차지하고 있다. SmartThings는 가정 자동화, 에너지 관리, 보안 등 다양한 영역에서 활용되고 있으며, 음성 비서, 인공지능 등 다른 기술과의 통합을 통해 그 활용 범위를 계속 확장하고 있다.

2. 실험 및 구현

전체 파이프라인은 데이터 생성, 모델 학습, Tapestry DHT 구성, SmartThings 통합의 4단계로 구성된다. 데이터 생성 단계에서는 다양한 가전 기기의 사용 데이터를 생성한다. 가전 기기의 평균 사용 시간을 명세서와 같이 입력하여 이를 초과하는 경우 이상 상태로 감지한다. 예를 들어 선풍기의 평균 사용 시간 2시간을 초과하거나 커피 포트 평균 사용 시간 1분을 초과하는 경우 이상 상태를 감지하도록 한다. 화재 위험 상태를 알릴 수 있도록 가전 기기의 전류 흐름이 읽히는 시점과, 평균 사용 시간 이상 시점인 임계 시점을 설정하여 각 시점의 데이터 패턴을 파일로 저장하고, 모델이 학습할 수 있도록 한다.

디바이스 모델의 번호를 부여하고 추후 파일의 변경 상태가 감지 될 경우 파이프라인을 통해 Smartthings 앱에 통지한다.

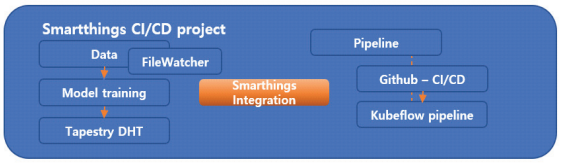
생성된 데이터를 바탕으로 딥러닝 모델을 학습한다. 모델은 시작 부분과 임계 영역 데이터를 학습할 수 있도록 두 개의 신경망 모델을 생성하여 가전 기기를 판별하고, 평균 사용 시간을 초과하는 경우 화재 위험을 감지한다. 모델은 다중 클래스 분류로 모델 학습 단계는 스크립트를 통해 이루어지며, 학습된 모델은 각각 시작과 임계 시점의 모델 파일로 저장된다.

Tapestry는 분산 시스템에서 객체(데이터)를 효율적으로 저장하고 검색할 수 있게 하는 DHT 기반의 알고리즘이다. 각 노드는 고유한 식별자를 가지며, 데이터를 효율적으로 검색하기 위해 라우팅 테이블을 사용한다. Tapestry 알고리즘을 활용하여 전류 및 정격 전압 값 데이터를 관리할 수 있도록 한다. K-means 클러스터링을 사용하여 데이터를 분할하고 Tapestry DHT 구조를 설정한다. Tapestry DHT를 통해 데이터를 분산 네트워크에 저장한다. 각 클러스터는 특정 패턴을 가진 데이터를 포함하고, 해당 클러스터의 노드는 이 데이

* 이 논문은 2024년도 정부 재원(과학기술정보통신부 여대학원생 공학연구팀제 지원사업)으로 과학기술정보통신부와 한국여성과학기술인육성재단의 지원을 받아 연구되었습니다. (WISET 제 2024-143호)

터를 관리한다. 데이터의 ID를 사용하여 Tapestry 노드 간의 라우팅 테이블을 통해 데이터를 효율적으로 검색할 수 있도록 한다. 전류 데이터를 포함하는 각 데이터 포인트는 특정 클러스터에 속하게 되며, 이는 전류 데이터의 패턴을 기반으로 태그된다.

SmartThings API와의 통합을 통해 모델 번호를 확인하고, 시간 초과 시 애플리케이션에 통보한다. 통합 구현 파일을 사용하여 이상 상태를 감지하고, 이상 상태 발생 시 SmartThings에 알림을 보낸다. 이를 통해 가전 기기의 상태를 실시간으로 모니터링하고, 이상 상태 발생 시 즉각적으로 대응할 수 있다.



(그림 1) 파일 구조도

데이터를 생성하여 실험하기 위해 5개의 가전 제품(선풍기, 커피포트, pc, 모니터, 헤어드라이어)에 대하여 기기별 사용 데이터 100개 세트를 생성한다. 선풍기는 평균 사용 시간 2시간으로 설정하고 2시간 초과 시 이상 상태를 감지하도록 한다. 커피포트는 평균 사용 시간 1분으로 설정하고 1분 초과 시 이상 상태를 보고하도록 한다. 디바이스 사용 분류 파일과 시작 시점, 임계 시점의 전류 흐름을 파일로 생성하여 저장한다. 데이터는 전원 켜진 시점부터 10초까지의 시작 데이터 패턴과 2시간 지점의 임계 데이터 패턴으로 지정하고 각 데이터 포인트에 모델 번호를 부여하여 저장한다. 데이터는 시작 부분과 임계 영역 데이터를 분리하여 전처리한다. 시간, 전압, 전류 등의 정보를 포함하며, 정상과 이상 상태의 여부를 표시하여 딥러닝 및 이상 감지에 사용할 데이터셋을 생성한다.

64-32개의 노드로 이루어진 ReLU 활성화 함수를 통해 각 기기에 대한 입력 데이터가 복잡한 패턴을 학습할 수 있도록 한다. 다섯 개의 노드로 이루어진 출력 레이어에 Softmax 활성화 함수를 사용하여 다중 클래스 분류를 수행하고, 예측값과 실제값의 차이를 계산하기 위한 Categorical Crossentropy 손실 함수를 통해 모델을 최적화한다. Adam 옵티마이저를 설정하여 빠르고 안정적인 학습을 지원할 수 있도록 한다. 신경망 모델의 구성을 통해 모델의 기기 사용 패턴을 분류한다.

Tapestry의 각 노드는 고유한 ID를 가지며, 라우팅 테이블을 통해 다른 노드와 연결된다. 파일에서 기기 사용 분류를 위한 파일의 데이터를 읽어와 사용 시간과 전압 데이터를 기반으로 KMeans 클러스터링을 수행하여 각 클러스터에 대해 Tapestry 노드를 생성하고 각 클러스터의 데이터를 해당 Tapestry 노드에 추가한다. 노드 ID와 라우팅 테이블을 초기화하고 새로운 노드를 라우팅 테이블에 추가하여 두 노드 ID 간의 공통 부분의 길이를 계산하여 주어진 키와 가장 일치하는 노드를 라우팅 테이블에서 찾는다. 랜덤한 16진수 문자열 ID를 생성하여 문자열로 반환한다.

본 논문에서는 Smartthings로의 알림을 네트워크 요청 및 개발 및 테스트 과정에서 API Rate 제한, API 토큰 사용, 네트워크 상태에 대한 제약으로 인해 Mock으로 Device ID를 표시하는 것으로 대체하였다. 실제 SmartThings API URL로 SmartThings API의 엔드 포인트를 정의하고 API 접근을 위한 인증 토큰을 설정하여 SmartThings API를 호출하여 특정 장치에 알림을 전송할 수 있다. 임시 데이터와 함수를 사용하여 비정상 상태를 감지하고 알림을 보내는 전체 프로세스를 시뮬레이션한다.

Kubeflow Pipelines를 사용하여 ML 파이프라인을 정의하여 여러 컴포넌트를 데이터 생성, 모델 학습, Tapestry DHT 설정, SmartThings 통합을 순서에 따라 실행할 수 있도록 한다. 각각을 컴포넌트로 정의하고 파이프라인을 정의하여 컴포넌트가 순서대로 실행되도록 한다. 파이프라인을 YAML파일로 컴파일하여 Kubeflow Pipelines에서 사용할 수 있도록 한다. 이 파이프라인은 데이터생성부터 모델 학습, Tapestry DHT 설정, SmartThings 통합까지 일련의 작업을 자동화하여 실행하는 기능을 한다.

watchdog 라이브러리를 사용하여 파일 시스템의 변경 사항을 모니터링하도록 한다. 특정 파일의 변경을 감지하고, 변경이 발생하면 Kubeflow 파이프라인을 재실행한다. 파일 시스템 변경 이벤트를 처리하는 핸들러 클래스를 정의하여 DataChangeHandler 클래스는 FileSystemEventHandler를 상속받아 on_modified 메서드가 파일이 수정될 때 호출되도록 한다. 파이프라인 스크립트를 실행하고 이 스크립트는 Kubeflow 파이프라인을 재실행하도록 한다.

(그림 2) Github Workflows[1]

빌드 및 배포의 자동화를 위하여 GitHub Actions는 .github/workflows 디렉토리에 저장된 YAML 파일을 통해 워크플로우 정의 파일을 자동으로 인식하고 실행한다. 자동화된 파이프라인을 사용하면 배포 과정이 일관되게 수행하도록 한다. GitHub 레포지토리에 변경 사항이 발생할 때 자동으로 실행되며, Kubeflow 파이프라인을 테스트하고, Docker 이미지를 빌드하여 Docker Hub에 푸시한 다음, Kubeflow에 파이프라인을 업로드하는 단계를 수행한다.

3. 결 론

본 연구에서는 SmartThings와 Kubernetes를 활용한 CI/CD 접근 방식을 통해 스마트홈 환경에서의 데이터 처리와 이상 감지 시스템을 제안하였다. Kubernetes와 Kubeflow를 이용한 자동화 파이프라인 구축을 통해 시스템의 효율적인 개발, 업데이트 및 배포를 지원하였으며, SmartThings와의 통합을 통해 실시간 모니터링 및 알림 기능을 구현하였다. 이를 통해 스마트홈 환경에서의 안전성과 효율성을 크게 향상시킬 수 있다.

참 고 문 헌

[1] 이승환, Kubernetes 기반 CI/CD 환경에서 무중단 자동화 배포 전략, 단국대학교 일반대학원, 2022

[2] Smartthings, https://developer.smartthings.com, SAMSUNG, Aug 2024

[3] Github Action, https://docs.github.com/en/actions/about-github-actions/understanding-github-actions, Aug 2024