Flask의 모델 서빙을 이용한 웹 어플리케이션 구현 : Urinary Stone 인공지능 응용

이충섭 [1], 임동욱 [1], 노시형 [1], 김지언 [1], 유영주 [1], 김태훈 [1,4], 박성빈 [3], 윤권하 [1,2], 정창원 [1,4]
[1] 원광대학교 의료융합연구센터
[2] 원광대학교 의과대학 영상의학과
[3] 중앙대학교 병원 영상의학과
[4] 원광대학교병원 정보관리실, 스마트사업팀

e-mail : { cslee99, dwl316, nosij123, kakasky112, yeriel5yj, tae_hoonkim}@wku.ac.kr, pksungbin@paran.com, {khy1646, mediblue}@wku.ac.kr

Web Application Implementation Using Flask Model Serving : Urinary Stone Artificial Intelligence Application

Chung-Sub Lee[1], Dong-Wook Lim[1], Si-Hyeong No[1], Ji-Eon Kim[1], Yeong-Ju Yu[1],Tae-Hoon Kim[1,4], Sung Bin Park[3], Kwon-Ha Yoon[1,2], Chang-Won Jeong[1,4]
[1]Medical Convergence Research Center, Wonkwang University
[2]Dept of Radiology, Wonkwang University School of Medicine and Hospital
[3]Department of Radiology, ChungAng University Hospital, Seoul, Republic of Korea
[4]Smart Business Team, Information Management of Wonkwang University Hospital

요 약

본 논문은 웹의 발달로 인하여 의료 서비스들이 기존의 Client-Server 방식의 제품에서 Web 방식의 제품으로 변경되고 있는 현대 흐름에서 인공지능 어플리케이션 또한 Web 으로 서비스 하기 위한 방법과 구현된 요로결석 AI 어플리케이션에 대해 기술한다. 이를 구현하기 위해 Python 기반의 Flask 라는 마이크로 웹 프레임워크를 사용하여 DICOM 핸들링, Pre-Processing, Mask 를 생성하고 Predict 결과를 Model Serving 을 통하여 Urinary Stone Segmentation Model 이 서비스되는 인공지능 웹 어플리케이션 동작 방식과 수행 결과를 보인다.

1. 서론

초창기의 1 세대 웹은 모두 정적(static)인 웹사이트였다. 2, 3 세대 웹에서 JavaScript 가 나오면서 동적(dynamic)인 웹사이트가 나왔다. 3 세대 웹부터 SPA(Single Page Application)[1]이라는 개념이 등장했다. SPA 는 기본적인 웹 어플리케이션에 필요한 모든 정적 리소스를 최초에 한번 다운로드 받고 이후 갱신에 필요한 데이터만 전달받아 페이지를 렌더링하므로 전체 페이지를 재 렌더링하지 않아 트래픽 감소와 네이티브 앱과 유사한 사용자 경험을 제공한다. SPA 형태의 웹이 나오면서 트래픽 감소, 네이티브 앱과 유사한 사용자 경험, 웹 기술의 발전에 따라 기존의 페이지 형태의 웹 개발에서는 할 수 없었던 네이티브앱 형태의 개발로 발전하게 되었다. 의료 분야의 시스템들 또한 방대한 양의 데이터를 이용하고 업무 특성상 속도에 민감한 시스템이어야 하기 때문에 기존의 Client-Server 방식의 Application 형태를 유지해왔지만 점차 웹의 발달과 더불어 웹 방식의 제품으로 변화하기 시작했다.

인공지능 연구 또한 기존의 학습을 통한 예측 모델을 개발하고 다양한 방법을 적용하여 예측 모델의 성능을 높이기 위한 연구에서 현재 운영되는 제품에 예측 모델을 적용하여 사업적 성과를 만들기 위한 다양한 방법을 시도하고 있다. 웹 방식의 제품으로 변화된 의료 서비스들과 연동하기 위해서 RestAPI 형태로 실제 운영 서비스에 적용하여 예측값을 사용자에게 전달하는 “model serving” 기술이 최근 주목 받고 있다[2].

본 논문에서는 Python 으로 개발된 마이크로 웹 프레임워크 중 하나인 Flask[3]를 사용하여 Urinary Stone Segmentation 모델을 웹 기반으로 서비스 하는 인공지능 웹 어플리케이션에 대해서 기술한다.

본 연구는 보건복지부의 재원으로 한국보건산업진흥원의 보건의료기술 연구개발사업(HI18C1216) 그리고 한국연구재단(NRF-2018R1D1A1B07048833)(NRF-2020R1I1A1A01074256) 지원에 의하여 이루어진 것임.

2. 관련 연구

기존 Model Serving 플랫폼은 목표와 능력에 따라

매우 다양하다. 예를들어, 일부 플랫폼은 지연 시간이 매우 짧은 유형이 있고, 다른 플랫폼들은 사용 편의성과 간단한 추론 인터페이스를 우선시 하는 플랫폼 등 다양하다. 실제 Model Serving 은 PennAI[4], Tensorflow Serving[5], Clipper[6], AWS Sagemaker[7], Flask Serving 등 여러 형태의 방법으로 연구되고 서비스화되고 있다. 특히, Flask 를 이용한 Serving 은 사용자가 Python 으로 개발하여 Model 을 생성할 때 사용했던 코드와 크게 다르지 않게 간단하게 RestAPI 를 구축할 수 있다는 장점이 있다. 표 1 은 Model Serving 플랫폼 간의 차이점을 아래와 같은 기준으로 분류했다.

각 플랫폼별 비교 항목은 아래와 같다.

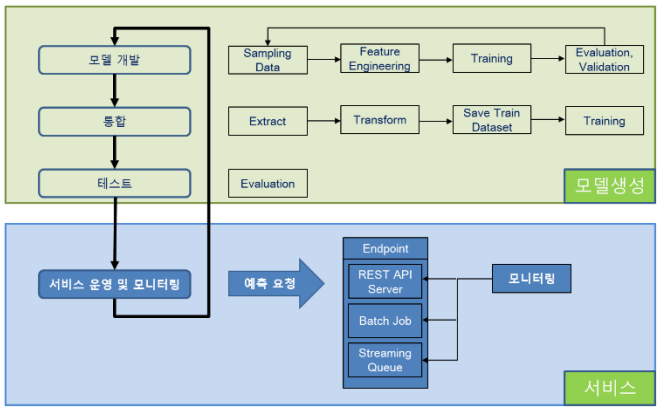
- •서비스 모델 : 호스팅 서비스 또는 셀프 서비스와 같이 제공되는지 여부
- •모델 유형 : 지원되는 언어 및 유형
- •입력 유형 : 지원되는 입력 유형의 범위
- •Train 여부 : 학습을 지원하는지 여부
- •변환여부 : Pre / Post-Processing 처리 여부
- •워크플로우 : 모델과 변환코드 간의 워크플로우 지원
- •호출 인터페이스 : 모델간의 호출 방법
- •실행 환경 : 모델이 배포되는 위치

<표 1> Model Serving 플랫폼 비교

	PennAI	TF Serving	Clipper	SageMaker	Flask
Service Model	Hosted	Self-service	Self-service	Hosted	Hosted
Model types	Limited	TF Servables	General	General	General
Input types supported	Unknown	Primitives, Files	Primitives	Structured, Files	Structured, Files
Training supported	Yes	No	No	Yes	Yes
Transformations	No	Yes	No	No	Yes
Workflows	No	No	No	No	Yes
Invocation interface	Web GUI	gRPC, REST	gRPC, REST	gRPC, REST	API, REST
Execution environment	Cloud	Docker, Kubernetes, Cloud	Docker, Kubernetes	Cloud, Docker	Docker, Kubernetes, Singularity, Cloud

3. 제안 시스템

일반적인 모델 개발과 서빙을 위한 워크플로우는 그림 1 과 같다.



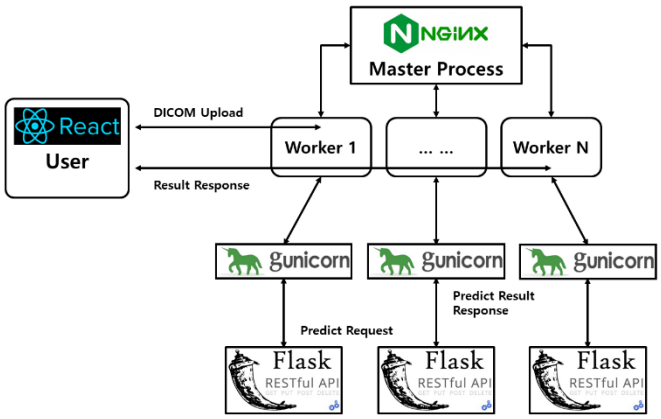
(그림 1) 인공지능 Model 개발 및 Model Serving

모델 생성 단계에서는 Urinary Stone 데이터를 수집하고 Wonkwang-J 라는 Labeling Tool 을 이용하여 Urinary Stone 에 대한 Mask 파일을 생성하였다. 학습을 위해서 수집한 DICOM 데이터와 정답으로 인식하기 위한 Mask 데이터를 Input 데이터로 입력하여 학습하였다. 학습을 위한 데이터는 Train 데이터 : Axial 영상 800 장 Mask 800 장, Validation 데이터 : Axial 영상 100 장 Mask 100 장, Test 를 위한 Axial 영상 200 장의 Dataset 을 준비하여 Train 과 Evaluation 을 진행하여 Model.json 과 Weight.h5 를 생성했다. 이렇게 생성한 Model 과 Weight 파일을 가지고 Predict 가 가능하지만 웹 서비스 형태로 서비스 하기 위해서 Model Serving 을 지원해야 했다. 따라서 여러 Model Serving 방법들 중에 Python 문법 기반으로 개발 가능한 Flask 라는 Python 기반의 마이크로 웹 프레임워크를 사용하여 직접 Model Serving 을 구현하였다.

우리는 Tensorflow 기반으로 서비스에 적용하기 위해서 개발한 인공지능 Model 을 기존의 학습 코드와 docker 를 이용하여 간단하게 serving 할 수 있다. 그러나 의료영상을 핸들링하는데 있어서 Pre-Processing 단계가 Tensorflow-serving 에서는 지원되지 않기 때문에 이를 해결해야만 했다. Flask 는 OpenCV, Numpy, Scikit-image 라는 패키지를 설치하여 DICOM 을 오픈하거나 이미지 Pre-Processing 및 다양한 영상처리 API 를 지원하고 있다. 그리고 Python 기반에서 할 수 있는 모든 기능이 웹에서 사용 가능하기 때문에 의료영상 기반의 인공지능 Model Serving 에 용이한 웹 프레임워크로 채택하여 소프트웨어 아키텍처에 적용하였다.

3-1. Model Serving 전체 Workflow

본 논문에서 제안하는 Flask 를 이용한 Urinay Stone 인공지능 웹 어플레이션의 전체 구조는 그림 2 와 같다.



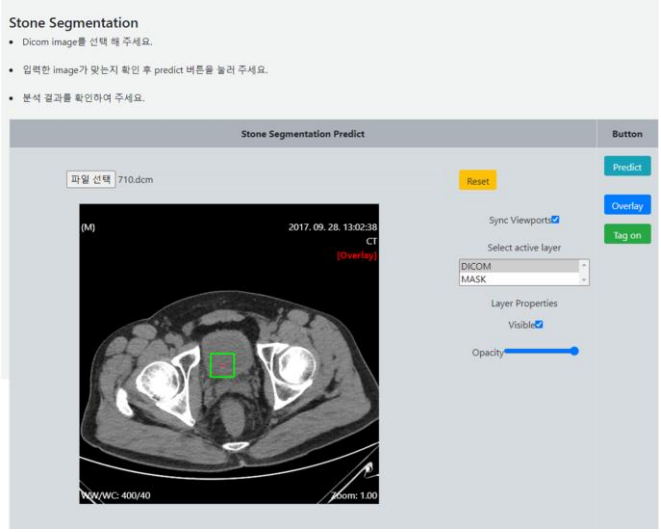
(그림 2) Model Serving Workflow

React 로 개발한 Front-End 에서 DICOM 을 업로드하면 Web Server 인 nginx 는 다양한 User 의 요청을 처리하기 위해서 Master Process 와 Worker Process 를 이용하여 부하를 감당한다. Master Process 는 설정 파일에 정의된 개수만큼 Worker Process 를 생성하고 Worker Process 사이에 요청을 효율적으로 분배하여 처리하도록 한다. 여기에서 nginx 와 Gunicorn

을 함께 사용하게 되면 동시에 많은 요청을 처리할 수 있고, 훨씬 안정화된 서버를 구축할 수 있다. 만약, Gunicorn 없이 flask 와 nginx 를 직접 연동하게 되면 flask 는 단일 프로세스, 단일 쓰레드이기 때문에 nginx 에서 동시에 많은 요청을 처리하더라도 결국 flask 내부에서 병목이 발생한다. 따라서 gunicorn 을 사용하여 worker process 들의 pool 을 관리하며 nginx 로부터 요청이 올때마다 적절한 할당을 하기 때문에 병렬적인 처리를 가능하게 한다.

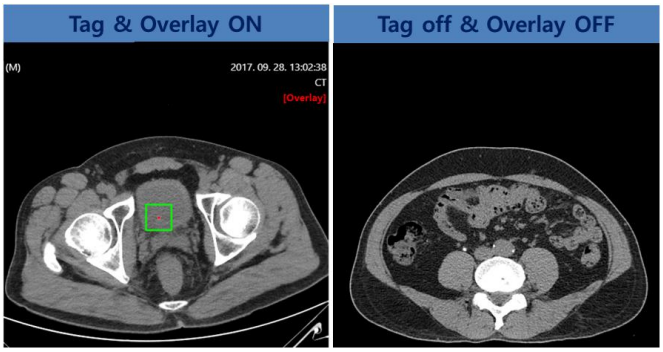
3-2. 인공지능 웹 어플리케이션

본 논문에서 제안하는 Flask 를 이용한 Urinary Stone 인공지능 웹 어플리케이션의 전체 UI 는 그림 3 과 같다.



(그림 3) Urinary Stone 인공지능 웹 어플리케이션

파일 선택 버튼을 클릭하여 병변이 예상되는 DICOM Image 를 단일 파일 또는 폴더 단위로 선택한다. 선택 후 DICOM 파일 업로드가 완료되면 Predict 버튼을 클릭하여 DICOM 파일을 프로그램에서 인식하기 좋도록 변경하는 Pre-Processing 단계를 거쳐서 Predict 요청을 한다. Predict 결과로 서버 저장소에 Urinary Stone 으로 인식되는 위치에 빨간색으로 Labeling 하여 Mask 파일을 생성한다. 하지만 Urinary Stone 은 대부분은 눈으로 식별 가능한 수준의 크기여서 큰 문제가 없지만 간혹 작은 크기도 검출이 되는데 이 부분은 빨간색으로 Labeling 을 하더라도 3㎜ 미만일 경우 식별하기 어렵다. 그래서 Urinary Stone 상하좌우에 일정 간격을 두고 녹색으로 Bounding Box 를 생성하여 Urinary Stone 이 잘 식별되도록 Mask 파일을 생성하였다. Reset 버튼은 선택한 DICOM 을 선택 해제할 때 사용한다. Sync Viewports 에 체크하게 되면 원본 DICOM Image 에 Mask 를 Overlay 후 windowing 기능, 또는 Zoom, Pan 기능을 사용할 때 원본이미지와 Mask 를 동시에 컨트롤하고자 할 경우 사용된다. Select active layer 는 체크박스에서 DIOCM 을 선택하면 DICOM 원본 영상만 보이고 MASK 를 선택하면 Mask 파일만 보인다. Layer Properties 는 Select active layer 에서 선택된 Layer 의 투명도를 조절할 수 있다. Overlay 버튼은 BoundingBox 와 Urinary Stone 영역에 대한 Mask 를 Visible, Invisible 처리할 수 있다. Tag on 버튼은 상단 좌우측에 DICOM Tag 정보를 Display 하고 있는데 해당 정보를 Visible, Invisible 처리할 수 있다. 그림 4 는 Overlay 버튼과 Tag on 버튼을 눌렀을 경우에 Display 의 변화를 확인할 수 있다.



(그림 4) Overlay 와 Tag on 수행 화면

4. 결론

본 논문에서는 기존 연구에서 생성된 Urinary Stone AI 모델과 Weight 파일로 Flask 로 개발된 웹 서비스를 통하여 Predict 할 수 있는 인공지능 웹 어플리케이션을 구현하였다. 개발된 어플리케이션을 이용하여 단일 파일, 멀티 파일에서 Urinary Stone 검출이 가능함을 보였고, 또한 한 파일 안에서 Multi Stone 도 검출 가능함을 보였다. 향후 연구내용으로는 해당연구를 진행하면서 요로가 아닌 곳의 아티팩트를 결석으로 검출하는 문제점과 3㎜ 이내의 Urinary Stone 의 경우와 촬영 조건에 따라 검출하지 못하는 케이스가 인공지능 알고리즘의 성능에 영향을 끼치고 있다. 이러한 문제점을 해결하기 위한 Pre-Processing 과 학습 파라미터 조정을 통해 정확도를 높이기 위한 연구를 수행할 계획이다.

참고문헌

[1] Single-page Application, https://en.wikipedia.org/wiki/Single-page_application
[2] Ryan Chard et al., “DLHub: Model and Data Serving for Science,” In 2019 IEEE International Parallel and Distributed Processing Symposium (IPDPS), pp. 283-292. IEEE, 2019.
[3] “Flask,” https://flask.palletsprojects.com/en/1.1.x/
[4] R. S. Olson et al., “A system for accessible artificial intelligence,” in Genetic Programming Theory and Practice XV. Springer, 2018, pp. 121–134.
[5] C. Olston et al., “TensorFlow-Serving: Flexible, high-performance ML serving,” in 31st Conf. on Neural Information Processing Systems, 2017.
[6] D. Crankshaw et al., “Clipper: A low-latency online prediction serving system,” in 14th USENIX Symposium on Networked Systems Design and Implementation (NSDI), 2017, pp. 613–627.
[7] “Amazon SageMaker,” https://docs.aws.amazon.com/sagemaker/latest/dg/whatis.html. Accessed October 14, 2018.