Journal of KIIT. Vol. 19, No. 10, pp. 115-123, Oct. 31, 2021. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2021.19.10.115

RFDSys: AIoT기반의 K-NN과 Motion Detection을 적용한 실시간 화재 검출 시스템

박종찬*, 강대성**

RFDSys: Real-time Fire Detection System using AIoT-based K-NN and Motion Detection

Jong-Chan Park*, Dae-Seong Kang**

이 논문은 2017 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No.2017R1D1A1B04030870)

요 약

최근 화재 발생 시 대피경로와 최적의 대응 시스템을 구축하기 위해 AIoT 기반 화재 판단을 위한 화재 추론 AI 디바이스 연구가 진행 중인 추세이다. 본 연구에서의 제안하는 방법은 기존의 실시간 검출 시스템에서 K-NN 배경 분리를 통해 배경과 전경을 분할한 뒤에 움직임이 발생하는 영역을 관심 영역으로 지정하여 해당 관심 영역에 화재 여부를 YOLOv4를 통해 추론하게 된다. 이때 배경의 영향을 줄이기 위해 데이터셋은 화염, 연기, 불꽃 이미지들을 검은색 이미지 위에 이미지 블랜딩하여 데이터셋을 생성하여 학습을 진행하였고, 화재 추론 AI 디바이스 Jetson 계열 Xavier NX에 학습된 YOLOv4와 TensorRT를 구축을 하였다. 제안하는 방법을 통해 얻은 성능 평가 결과 평균 추론 정확도 화염 95.94%, 연기 86.56%, 불꽃 92.3%와 추론 속도 평균 20.5 FPS로 화재를 검출할 수 있었다.

Abstract

Recently, fire inference AI device research is underway for AIoT-based fire judgment in order to establish an evacuation route and optimal response system in the event of a fire. In this work, the proposed method divides the background and foreground through K-NN background separation in the existing real-time detection system, and then designates the region where the motion occurs as the region of interest, inferring fire or not through YOLOv4. To reduce the impact of the background, the dataset created a dataset by blending the flame, smoke, and flame images onto the black image, and built the learned YOLOv4 and TensorRT on the fire inference AI device Jetson family Xavier NX. The performance evaluation obtained through the proposed method allowed fire detection with 95.94% mean inference accuracy flame, 86.56% smoke, 92.3% flame, and an average inference speed of 20.5 FPS.

Keywords

k-nearest neighbors, image blending, nvdia xavier nx, data augmentation, yolov4, tensorrt, rtsp camera

* 동아대학교 전자공학과 석사과정
- ORCID: https://orcid.org/0000-0002-1886-534X
** 동아대학교 전자공학과 교수(교신전자)
- ORCID: https://orcid.org/0000-0003-0186-2430

· Received: Jul. 21, 2021, Revised: Aug. 27, 2021, Accepted: Aug. 30, 2021

· Corresponding Author: Dae-Seong Kang
Dept. of Dong-A University, 37 NaKdong-Daero 550, beon-gil saha-gu, Busan, Korea,
Tel.: +82-51-200-7710, Email: dskang@dau.ac.kr

Ⅰ. 서 론

최근 노후화된 건물과 복잡하고 급변한 환경으로 인해 화재에 취약한 지역의 안전 관리에 어려움이 있다. 특히나 다세대 주택과 노령인구가 많은 지역에 화재 발생이 해마다 증가하여 인명피해가 발생하고 있다. 이를 해결하기 위해 AIoT(Artificial Intelligence of Thing)[1] 기반 화재 판단을 위한 연구를 진행 중이다. 이전 논문인 화재 예방 감시 시스템의 YOLO를 기반으로 한 실시간 화재 영상 검출에서 화재 데이터 생성을 통하여 YOLOv4에 학습후 AI 디바이스인 NVIDIA Jetson Nano에 YOLOv4를 구축한 뒤에 화염 추론을 하였다.

그림 1과 같이 MAP(Mean Average Precision)는 71.6%, FPS(Frame Per Second)는 1.2로 낮은 성능을 보였다. 또한 화염의 크기가 줄어들수록 추론 정확도는 더욱더 성능이 낮아지는 문제를 보였다[2].

본 연구는 이러한 문제점들을 해결하기 위해 기존의 시스템에서 전처리 기법을 적용하여 화재 추론 정확도를 개선하였다. 또한, 추론 속도를 개선하기 위해 이전 연구에서 사용된 AI 디바이스인 Jetson Nano에서 그림 2와 같이 Jetson 계열 Xavier NX를 선정하여 개발하였다.

NVIDIA Jetson Nano와 Jetson Xavier NX의 하드웨어와 사양을 비교해보면 Xavier NX에는 냉각 장치가 설치되어 있지만 Jetson Nano에는 방열판만 있으므로 실시간으로 화재 영상 검출을 하기 위해 Jetson Nano에서 YOLOv4를 적용하여 화재 추론을 하게 되면 저전력으로 운영하더라도 시간이 지날수록 하드웨어 발열이 높아져 셧다운이 되는 문제가 있다.

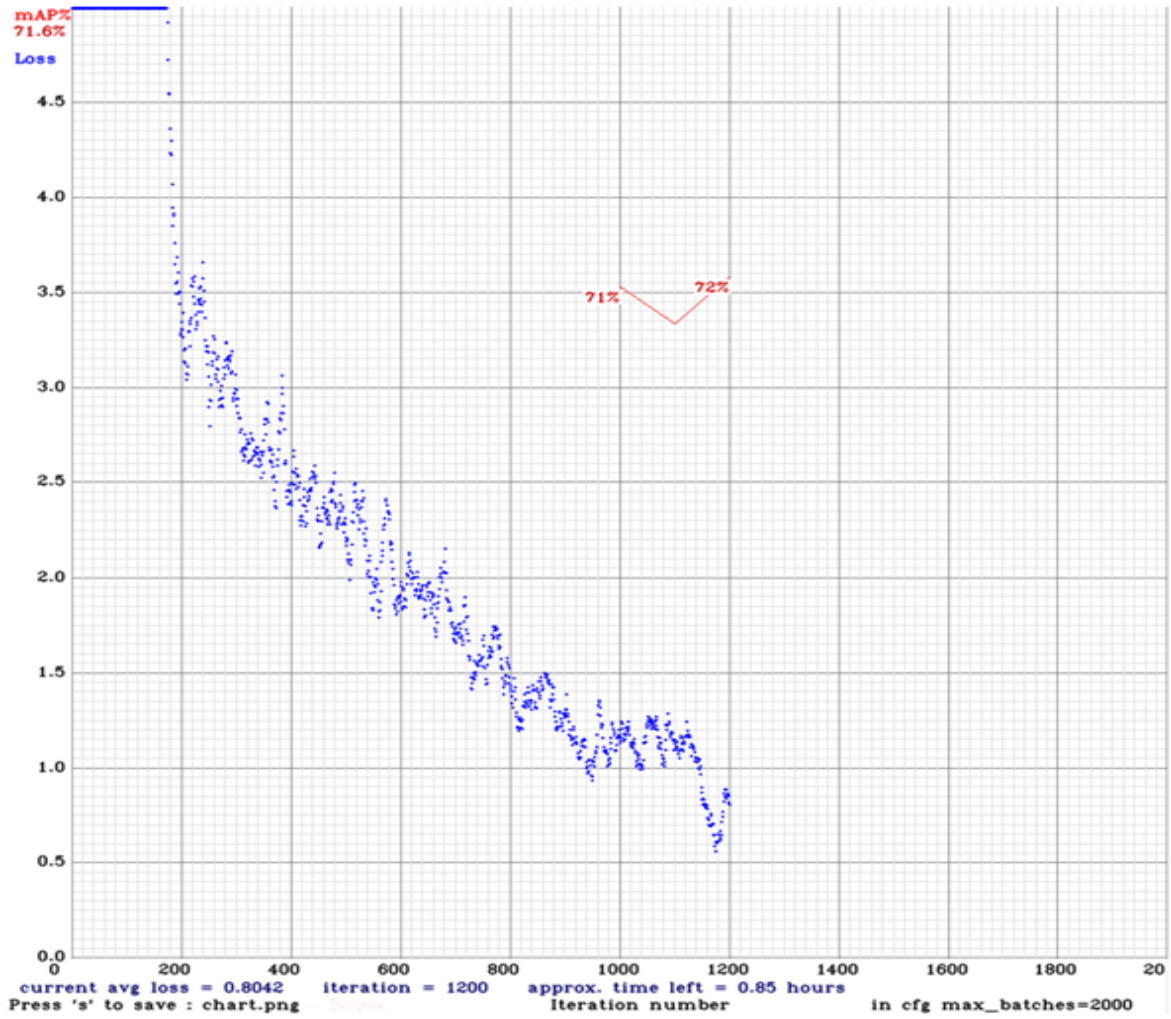

그림 1. 이전 논문의 화재 데이터셋을 학습하여 얻은 YOLOv4의 성능 지표

Fig. 1. Performance indicators of YOLOv4 obtained by learning fire datasets from previous papers

그림 2. NVIDIA Xavier NX 개발자 키트
Fig. 2. NVIDIA Xavier NX developer kit

또한, Xavier NX에는 48개의 Tensor 코어와 전용 뉴럴 네트워크 가속기가 있어 딥러닝 모델의 FPS 및 추론 성능을 크게 향상시킬수 있다[3].

즉, Jetson Nano에서의 문제를 해결하기 위해 Xavier NX를 활용하여 추론 AI 디바이스 개발을 하고 학습된 모델 최적화 도구를 적용하여 AI 디바이스 프로세서용으로 추론 모델 최적화를 한 뒤에 영상 전처리 기법을 적용하여 최적화된 모델 데이터를 로딩하여 실시간 화재 추론 성능을 개선하는 것이 본 연구의 핵심이다.

본 논문의 구성은 다음과 같다. 관련 연구에서는 YOLOv4, K-NN, 데이터 증강, TensorRT 기술들에 대하여 이론을 기술한다. 제안하는 방법에서는 이미지 블랜딩 기법, 전처리 기법, 데이터 증강 기법, TensorRT를 통해 화재를 추론하는 흐름에 대해서 기술한다. 실험방법 및 성능평가에서는 블랜딩 기법 및 전처리 기법을 적용하여 얻은 성능 지표와 영상 내에서 화염, 연기, 불꽃들의 추론 결과에 대해 논한다. 마지막으로 결론에 대해 기술하고 본 연구가 가지는 한계점 및 향후 연구 방안에 대하여 기술한다.

II. 관련 연구

2.1 YOLO(You Only Look Once)v4

객체 탐지 분야에서 많이 알려진 YOLO는 이미지가 주어지면 배경과 사물을 구분하고 어떤 사물을 인지하는 것을 말한다. 현재 YOLO는 YOLOv2, YOLOv3, YOLOv4, YOLOv5까지 다양한 버전이 나온 상태이다. 본 연구에서는 실시간 화재 추론을 하기 위해 YOLOv4를 사용하였다. YOLOv4는 그림 3과 같은 구조를 가지고 있으며 YOLOv3를 기반으로 CSPDarknet53, SPP(Spatial Pyramid Pooling), PAN(Path Aggregation Network) 기법들을 적용하여 성능을 개선한 모델이다[4][5].

2.2 K-NN(K-Nearest Neighbors) 배경 분리

K-NN는 분류나 회귀에 사용되는 비모수 방식이다. 새로운 데이터가 들어왔을 때 기존 데이터 그룹 중 어떤 그룹에 속하는지 분류 작업을 하는 알고리즘이다. 데이터로부터 거리가 가까운 k개의 다른 데이터의 레이블을 참조하여 분류하는 것이다.

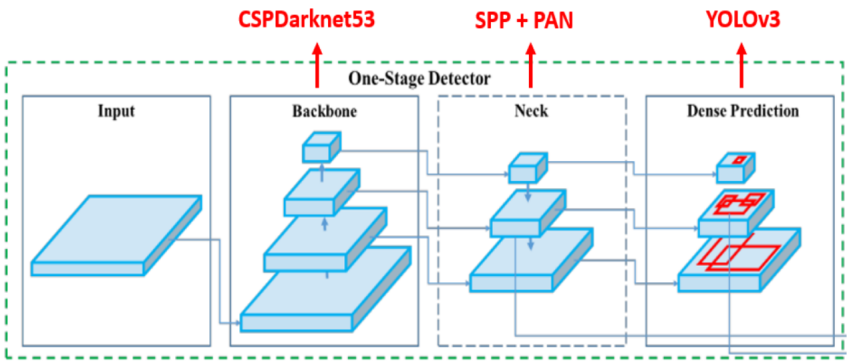

그림 3. YOLOv4의 구조
Fig. 3. Structure of YOLOv4

K-NN 배경 분리는 영상 전처리 기법 중에 하나이다. CCTV와 같은 고정된 카메라를 이용해서 영상을 받아오는 경우 배경이 되는 부분은 움직이지 않은 정지 영상이 될 것이고 움직이는 부분은 따로 추출하여서 배경 제거가 된다. 그림 4와 같이 현재 프레임과 배경 모델의 차를 통해 나온 정보를 임계값(threshold)을 조절하여 전경을 추출하는 방식이다.

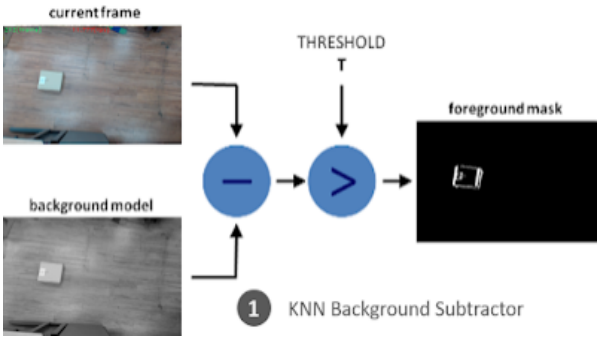

그림 4. K-NN 배경 분리 방법
Fig. 4. K-NN background subtractor method

본 논문에서는 K-NN를 적용하여 배경 분리를 하였다. YOLOv4에서 화재 검출을 할 시에 배경에 영향을 받기 때문에 전처리를 위하여 배경 분리를 하였다[6].

2.3 데이터 증강

데이터 증강은 데이터셋을 불러오고 모델을 학습하기 전에 전처리 기법 중의 하나이다. 일반적으로 광도 왜곡(Photometric distortions)과 기하학적 왜곡(Geometric distortions) 두 가지 데이터 증강 방법이 있다.

광도 왜곡에서는 이미지의 밝기, 대비, 색도, 채도 및 노이즈를 조정한다. 기하학적 왜곡은 임의 크기 조정, 자르기, 뒤집기 및 회전을 추가한다. 본 연구에서는 화재가 발생 시에 화염, 불꽃, 연기의 모양 및 크기가 불규칙하여 추론 정확도의 성능이 크게 떨어지므로 데이터 증강 기법을 통해 추론 정확도의 성능을 높이고자 하였다[7].

2.4 TensorRT

TensorRT는 학습된 딥러닝 모델을 최적화하여 NVIDIA GPU 상에서의 추론 속도를 수배, 수십 배까지 향상시켜주는 모델 최적화 엔진이다.

TensorRT는 NVIDIA GPU 연산에 적합한 최적화 기법들을 이용하여 모델을 최적화하는 Optimizer와 다양한 GPU에서 모델 연산을 수행하는 Runtime Engine을 포함한다[8].

본 연구에서는 TensorRT을 적용하기 위해 YOLOv3는 모델을 바로 적용할 수 있지만, YOLOv4의 경우 기존 모델과 구성이 달라서 몇 가지 변환을 통해서 모델을 TensorRT로 변환해야 한다. 변환하기 위해 그림 5와 같이 첫 번째로 DarkNet 모델을 ONNX(Open Neural Network Exchange) 모델로 변환해야 합니다. ONNX는 Tensorflow, Pytorch와 같은 서로 다른 DNN 프레임 워크 환경에서 만들어진 모델들을 서로 호환되게 사용할 수 있도록 만들어진 공개 플랫폼이다[9].

두 번째로 변환된 ONNX[10]모델을 TensorRT모델로 변환한다.

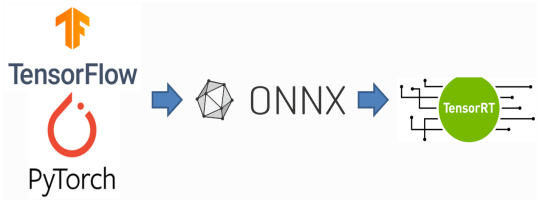

그림 5. TensorRT모델로 변환 과정
Fig. 5. Process for converting to a TensorRT model

III. 제안하는 방식

본 연구에서 사용된 화재 데이터셋의 종류는 그림 6과 같이 화염, 연기 그리고 불꽃 데이터셋을 이미지 블랜딩을 통하여 생성 하였다.

(a) 화염 (b) 연기 (c) 불꽃
그림 6. 화재 검출을 위한 데이터셋 종류
Fig. 6. Dataset types for fire detection, (a) flame, (b) smoke, (c) spark

이미지 블랜딩을 할 때에 배경 이미지는 검은색을 사용하였다. 특정 배경 이미지 위에 화염, 연기, 불꽃 이미지들을 블랜딩 하게 되면 특정 환경에서는 추론 정확도 성능이 좋으나 환경이 달라지면 추론 정확도가 떨어지기 때문이다.

생성된 데이터셋을 YOLOv4에 학습을 하고 본 연구에서 사용된 화재 추론을 위한 전체 흐름도는 그림 7과 같다.

화염 영상, 연기 영상, 불꽃 영상들로부터 들어오는 입력 영상을 K-NN 배경 분리 알고리즘을 적용하여 배경과 전경 분할을 한 뒤에 이진 영상에 기본적인 모폴로지 연산중 하나인 팽창 연산을 통해 잡음을 제거하고, 윤곽선을 검출한 뒤, 면적이 임계값 이상의 화소 위치를 움직임이 있는 화소로 판단하여 좌표 추출을 한 뒤에 관심 영역으로 지정하여 학습된 YOLOv4에 관심 영역 정보를 보내줌으로써 관심 영역에 해당하는 부분만 검출하게 된다.

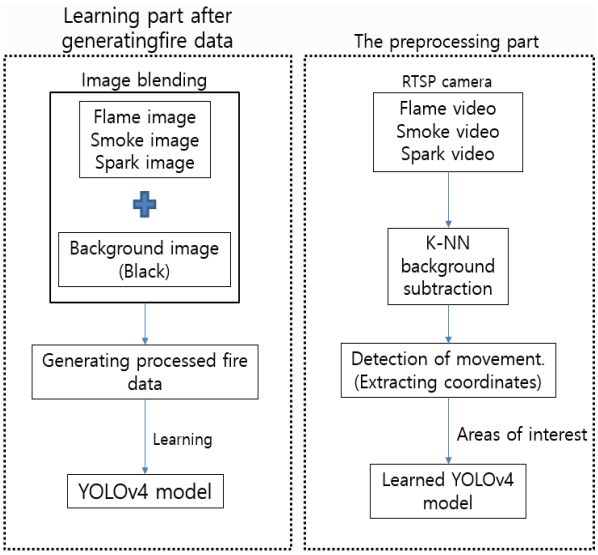

그림 7. 화재 검출을 위한 제안하는 과정
Fig. 7. Proposed process for fire detection

3.1 Data Augumentation을 적용한 YOLOv4학습

생성된 화재 데이터셋을 데이터 증강 기법을 적용하여 YOLOv4에 학습을 하기 위해 우선 YOLOv4에 있는 데이터 증강 기법인 angle을 30으로 맞추어 이미지를 30도 돌리면서 학습을 하도록 설정한다.

추가적으로 실재 화재 추론 테스트를 하였을 때 화재의 모양과 크기가 다르고 거리가 멀어질수록 추론 정확도가 떨어지는 걸 알수 있었다. 이러한 문제를 개선하기 위해 데이터 증강 기법 중 하나인 데이터셋 크기를 줄이고 늘리는 기법을 적용하여 학습을 진행하였다. 이미지 크기를 줄이면 작은 객체를 검출하는데 추론 정확도가 증가하고 이미지가 늘어나 커지면 큰 이미지에 대한 추론 정확도가 증가한다. 화재 데이터셋 이미지 크기는 416x416이지만 학습을 하는 동안에 352x352, 384x384, 416x416, 508x508, 544x544, 576x576의 크기들로 줄이고 늘리면서 학습을 진행하였다.

두 가지의 데이터 증강 기법을 적용하여 YOLOv4에 화재 데이터셋을 학습을 하였다. 화염, 연기, 불꽃 추론을 위한 작업 흐름은 그림 8과 같다.

3.2 학습된 YOLOv4을 Xavier NX에 구축 및 TensorRT로 변환

Xavier NX에 학습된 화재 YOLOv4의 환경을 구축을 한다. 이후 화재 YOLOv4를 ONNX로 변환하여 ONNX 파일이 생성을 한 뒤에 생성된 ONNX를 다시 TensorRT 파일로 변환을 한다.

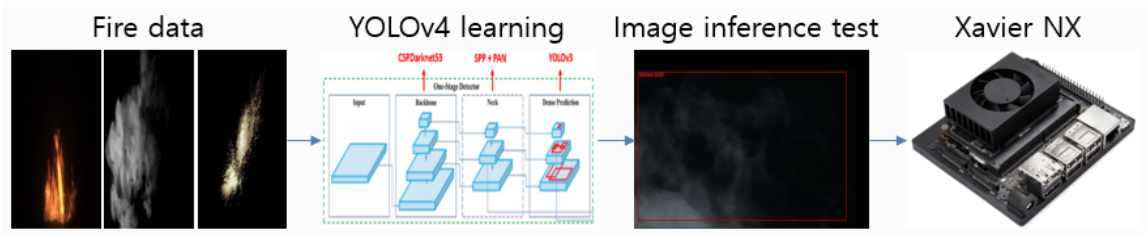

그림 8. 화재 추론을 위한 흐름
Fig. 8. Flow for fire inference

3.3 RTSP카메라 / Xaiver NX와 연결

최종 실시간으로 화재 영상 검출을 하기 위해 RTSP 카메라를 포트포워딩을 통해 Xavier NX와 연결을 한다. 그림 9와 같이 화염, 연기, 불꽃들이 RTSP 카메라를 통해 영상이 들어오면 Xavier NX에서 YOLOv4를 통해 추론을 한 뒤에 추론된 결과를 모니터로 전송을 하여 화재 검출 여부를 판단을 한다[11].

제안하는 방법을 통해 추론 정확도와 추론 FPS의 성능이 크게 향상된 것을 확인하였다. 연기나 불꽃의 테스트는 출력된 이미지를 통하여 실시간으로 추론 테스트를 하였다.

IV. 실험방법 및 성능평가

이미지 블랜딩을 통하여 생성된 화재 데이터를 YOLOv4에 학습을 시키기 위한 환경은 CPU는 Intel(R) Xeon(R) CPU @ 2.30GHZ, GPU는 NVIDIA Tesla K80, RAM은 25.51G 그리고 추론을 위하여 사용된 AIoT 디바이스 NVDIA Jetson Xavier NX는 CPU는 6-core NVIDIA Carmel 64-bit ARMv8.2 @ 1400MHz, GPU는 384-core NVIDIA Volta @ 110MHz with 48 Tensor Cores, RAM은 8G의 환경에서 실험을 하였다.

제안된 방법을 통한 화염, 연기 그리고 불꽃 검출 결과의 객관적인 평가를 위해 Eq. (1)의 정확도, Eq. (2)의 정밀도 그리고 Eq. (3)을 통해 에러율을 구하였다[12].

$$Accuracy = \frac{TP + TN}{TP + FN + FP + TN} \quad (1)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (2)$$

$$Error\ rate = \frac{FN + FP}{N + P} \times 100\% \quad (3)$$

TP(True positive)는 관심 영역 내에서 화재가 존재하는 경우에 YOLOv4에서 올바르게 판단한 횟수, FN(False Negative)은 화재가 존재하지만 YOLOv4에서 화재가 아닌 것으로 잘못 판단한 경우에 해당된다. FP(False Positive)는 화재와 관련이 없는 영상에서 YOLOv4가 화재로 잘못 판단한 경우, TN(True Negative)은 화재가 아닌 것으로 올바르게 판단한 경우이다.

표 1, 표 2, 표 3은 제안하는 방법을 통해 화재 영상으로부터 움직임을 추론하여 화염, 연기, 불꽃의 각 3가지의 비디오 영상을 통해 평균 정확도, 정밀도, 에러율, FPS의 결과를 통해 추론 성능을 확인하였다.

표 1. 화염 비디오 영상을 통한 화재 추론 성능 결과
Table 1. Fire inference performance results from flame video images

Flame video	Accuracy(%)	Precision(%)	Error rate(%)	FPS
Video 1	89.82	83	10.18	19.32
Video 2	98.83	94	1.17	19.67
Video 3	99.17	94	0.83	19.36

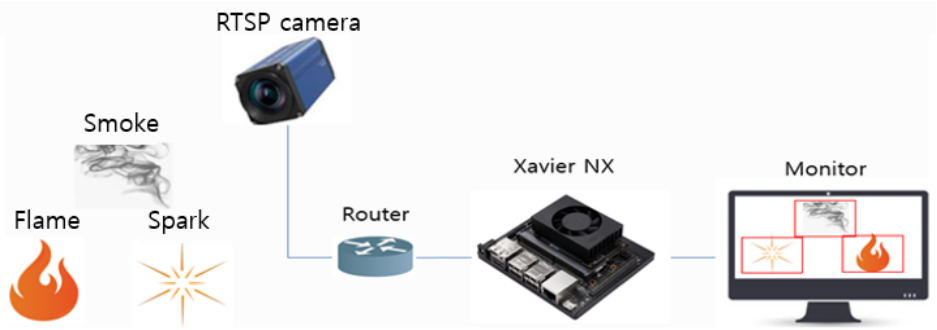

그림 9. 실시간 화재 검출 시스템
Fig. 9. Real-time fire detection system

표 2. 연기 비디오 영상을 통한 화재 추론 성능 결과
Table 2. Fire inference performance results from smoke video images

Flame video	Accuracy(%)	Precision(%)	Error rate(%)	FPS
Video 1	86.39	87	13.61	20.10
Video 2	82.40	85	17.60	19.91
Video 3	90.93	94	9.07	21.01

표 3. 불꽃 비디오 영상을 통한 화재 추론 성능 결과
Table 3. Fire inference performance results from spark video images

Flame video	Accuracy(%)	Precision(%)	Error rate(%)	FPS
Video 1	92.42	87	17.58	19.89
Video 2	93.12	94	6.88	19.51
Video 3	91.33	94	8.67	20.89

제안하는 방법을 통해 추론 성능 결과를 확인한 결과 화염 영상에서의 화염 정확도 95.94%, 정밀도 90.3%, 에러율 4.06%, FPS 19.45의 결과를 얻었고, 연기 영상에서의 정확도 86.56%, 정밀도 88.7%, 에러율 13.43%, FPS 20.34 그리고 불꽃 영상에서의 불꽃 정확도 92.3%, 정밀도 91.7%, 에러율 11.04%, FPS 21의 결과를 보여주었다. 연기의 경우 화염/불꽃의 비해 상대 적으로 낮은 정확도, 정밀도를 보였다. 본 연구에서는 화염, 연기, 불꽃 3가지의 화재를 검출 가능하므로 초기에 화재 추론이 가능하다는 장점이 있다.

그림 10은 제안하는 방법을 통해 K-NN 배경 분리 후 움직이는 영역을 관심 영역으로 추출하여 해당 영역의 좌표를 학습된 YOLOv4에 정보를 넘겨 화재 여부를 추론한 결과를 나타낸다.

그림 11, 12, 13은 화염, 연기, 불꽃들의 각 비디오 영상 3가지를 제안된 방법으로 화재를 추론하여 얻은 결과이다. 각 비디오 영상들은 실내외에서 촬영된 영상들을 수집한 것이며 화염 영상에서는 라이터 불꽃, 산불, 모닥불 그리고 연기 영상에서는 담배 연기, 화재 연기 마지막으로 불꽃 영상에서는 주로 불꽃 막대에서 나오는 불꽃들에 대한 영상들이다.

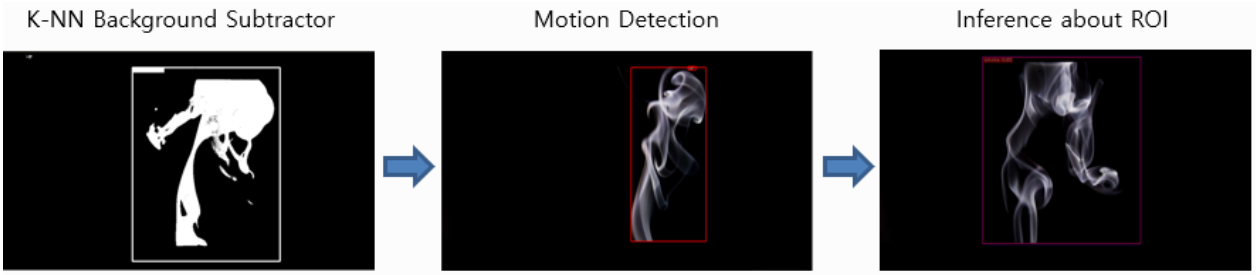

그림 10. 비디오 영상에서 K-NN 배경 분리 후 움직이는 영역 검출 후 관심 영역을 YOLOv4로 추론 결과
Fig. 10. Inferring the region of interest to YOLOv4 after detecting the moving region after separating the K-NN background from the video image

그림 11. 화염 비디오 영상을 통한 화재 추론 결과
Fig. 11. Fire inference results from flame video

그림 12. 연기 비디오 영상을 통한 화재 추론 결과
Fig. 12. Smoke inference results from flame video

그림 13. 불꽃 비디오 영상을 통한 화재 추론 결과
Fig. 13. Spark inference results from flame video

표 4는 YOLOv4에서 Jetson nano와 GPU, Xavier NX와 GPU, TensorRT가 적용된 Xavier NX의 FPS 성능 지표를 비교하여 각각 나타내었다.

표 4. 평균 추론 속도 성능 지표(FPS)
Table 4. Average inference rate performance indicator (FPS)

AI device	FPS
Jetson nano & GPU	1.2
Xavier NX & GPU	5.2
Xavier NX & TensorRT	20.5

V. 결론 및 향후 과제

본 논문은 이미지 블랜딩 기법을 통하여 신뢰성 높은 화염, 연기, 불꽃 3가지의 화재 데이터셋을 생성하고 데이터 증강 기법을 YOLOv4에 적용을 한 후 학습을 한 뒤에 제안된 방법으로 전처리를 적용하여 화염 정확도 95.94%, 정밀도 90.3% 연기 정확도 86.56%, 정밀도 88.7% 불꽃 정확도 92.3%, 정밀도 91.7%의 결과를 얻었다. 초기 연구에서는 화염과 불꽃에 대한 추론 성능이 뛰어났으나 연기에 대한 추론 성능이 크게 떨어져 제안된 방법을 통해 추론 정확도가 크게 개선되는 결과를 보았다.

검출 속도에서도 학습된 YOLOv4를 화재 추론 AI 디바이스 Xavier NX에 구축을 한 뒤에 TensoRT를 적용하여 추론 속도를 비교해 본 결과 Jetson Nano & GPU는 FPS 1.2, Xavier NX & GPU는 FPS 5.2, Xavoer NX & TensorRT에서는 FPS 20.5의 결과를 얻어 TensorRT를 적용하였을 때 기존의 GPU만을 사용하였을 때 보다 YOLOv4 추론 속도가 크게 개선된 결과를 얻어 실시간 화재 영상 검출에 적합하다는 것을 보여주었다.

향후 이미지 블랜딩을 통해 더 많은 데이터셋을 생성을 하여 학습을 하여 추론 정확도 성능을 높이고 낮은 추론 정확도를 보이는 연기에 대하여 전이학습과 추가적으로 신뢰성 높은 연기에 대한 데이터셋을 생성하여 학습을 하여 추론 성능을 높이는

연구가 필요할 것으로 보인다.

Acknowledgement

"2021년도 한국정보기술학회 종합학술대회에서 발표한 논문(Real-time video fire detection based on YOLO in antifire surveillance systems)을 확장한 것임"

References

[1] Y. K. Kwon, S. Y. Kim, "Trend Analysis of IoT Technology Using Open Source", The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 20, No. 3, pp. 65-72, Jun. 2020. http://dx.doi.org/10.7236/JIIBC.2020.20.3.65.

[2] J. C. Park and D. S. Kang, "Real-time video fire detection based on YOLO in antifire surveillance systems", Proceedings of KIIT Conference, Jeju, Korea, pp. 179-181, Jun. 2021.

[3] NVIDIASolutionsArchitect, https://blogs.nvidia.co.kr [accessed: Jun. 02, 2021]

[4] Redmon. J. and Farhadi A., "YOLOv3: An Incremental Improvement", Computer Science, Apr. 2018. https://arxiv.org/abs/1804.02767.

[5] Bochkovskiy. A., Wang. C. Y., and Liao. H. Y. M., "YOLOv4: Optimal Speed and Accuracy of Object Detection", Computer Science, Apr. 2020. https://arxiv.org/abs/2004.10934.

[6] J. M. Lee and D. S. Kang, "Improving Real-time Fire Detection Using Designation of Fire Detection Area Through Background Subtraction and Adaptive Color SpaceFiltering According to Brightness Distribution", The Journal of Korean Institute of Information Tenchnology, Vol. 18, No. 4, pp. 51-57, Apr. 2020. http://dx.doi.org/10.14801/jkiit.2020.18.4.51.

[7] Kate, https://jetsonaicar.tistory.com/66. [accessed: Jun. 10, 2021]

[8] NVIDIA Data Scientis, https://blogs.nvidia.co.kr [accessed: Jun. 02, 2021]

[9] GoodYunMorning, https://yunmorning.tistory.com/17 [accessed: Jun. 11, 2021]

[10] D. H. Chang, J. S. Lee, J. Y. Heo "Lightweight of ONNX using Quantization-based Model Compression", The Journal of The Institute of Internet, Broadcasting and Communication, Vol. 21, No. 1, pp. 93-98, Feb. 2021. http://dx.doi.org/10.7236/JIIBC.2021.21.1.93.

[11] T. N. Viet, H. Q. Cong, and T. P. Minh, "Video Smoke Detection For Surveillance Cameras Based On Deep Learning In Indoor Environment", 2020 4th International Conference on Recent Advances in Signal Processing, Telecommunications & Computing (SigTelCom), Hanoi, Vietnam, pp. 82-86, Sep. 2020. https://doi.org/10.1109/SigTelCom49868.2020.9199056.

[12] S. Sergio, E. Abdussalam, and G. Alessio, "Real-time video fire/smoke detection based on CNN in antifire surveillance systemt", Journal of Real-Time Image Processing, pp. 889-900, Nov. 2020. https://link.springer.com/article/10.1007/s11554-020-01044-0.

저자소개

박 종 찬 (Jong-Chan Park)

2020년 2월 : 동아대학교 전자공학과(공학사)
2020년 3월 : 동아대학교 전자공학과 석사과정
관심분야 : 영상처리, 인공지능, 디지털 트윈

강 대 성 (Dae-Seong Kang)

1994년 5월 : Texas A&M 대학교 전자공학과(공학박사)
1995년 ~ 현재 : 동아대학교 전자공학과 교수
관심분야 : 영상처리, 인공지능, 디지털 트윈, 패턴인식