Faster-RCNN을 이용한 PCB 부품 인식

기철민* · 조태훈*

*한국기술교육대학교

Recognition of PCB Components Using Faster-RCNN

Cheol-min Ki* · Tai-Hoon Cho*

*Korea University of Technology and Education

E-mail : kigoon@koreatech.ac.kr

요 약

현재 딥러닝을 이용한 연구들이 활발하게 이뤄지고 있고, 많은 분야에서 좋은 결과를 보여주고 있다. PCB(Printed Circuit Board) 기판 위에 실장 된 부품을 인식할 때 템플릿 매칭을 이용한 방식이 주를 이룬다. 하지만 템플릿 매칭은 모양과 방향, 밝기에 따라 여러 템플릿이 존재해야하고, 영상 전체를 탐색하여 매칭하기 때문에 수행시간이 오래 걸린다. 또한 인식률이 상당히 떨어지는 단점이 존재한다. 이로 인해 본 논문에서는 하나의 영상에서 여러 개의 물체를 분류할 때 사용하는 기계학습 방법 중 하나인 Faster-RCNN(Region-based Convolutional Neural Networks)을 이용하여 PCB 부품들을 인식하는 방식을 사용하였으며, 이 방법은 템플릿 매칭 방식보다 수행시간과 인식 면에서 더욱 좋은 성능을 보여준다.

ABSTRACT

Currently, studies using Deep Learning are actively carried out showing good results in many fields. A template matching method is mainly used to recognize parts mounted on PCB(Printed Circuit Board). However, template matching should have multiple templates depending on the shape, orientation and brightness. And it takes long time to perform matching because it searches for the entire image. And there is also a disadvantage that the recognition rate is considerably low. In this paper, we use the Faster-RCNN method for recognizing PCB components as machine learning for classifying several objects in one image. This method performs better than the template matching method, execution time and recognition.

키워드

Machine Learning, Deep Learning, Faster-RCNN, Object Detection

Ⅰ. 서 론

현재 응용분야에서 인공지능(AI) 혹은 딥 러닝을 이용한 연구가 활발하게 이루어지고 있고, 비젼 장비를 이용한 PCB 기판 검사 분야에서도 적용이 되고 있다.

딥 러닝의 다양한 방법들 중 컴퓨터 비젼 분야에서 많이 사용되는 방법 중 하나는 Convolutional Neural Network(CNN)[1,2]이다. CNN은 일반적인 Multi-Layer Neural Network에 Convolution Feature를 추출하는 Convolution Layer가 추가된 형태이며, 기존의 Multi-Layer Neural Network에 비해 성능 향상과 파라미터를 상당히 줄일 수 있었다.

그러나 보통 CNN은 하나의 영상에서 하나의 대응 값만을 추측하는데, 하나의 커다란 영상에서 여러 개의 물체를 검출할 때 CNN을 사용하며 Region-Based CNN(RCNN) 류의 알고리즘을 사용한다.

그 중 Faster-RCNN[3]은 Region Proposal Network라는 특수한 망을 추가하여 이전에 연구된 RCNN류 알고리즘에 비해 속도와 성능의 향상을 이루었다.

본 논문에서는 PCB에 실장 된 부품을

Faster-RCNN을 이용하여 PCB에 실장 된 부품을 검출하는 방식을 제안하며, 기존에 PCB 분야에서 많이 사용되던 템플릿 매칭 방식과의 성능 및 속도 비교를 진행하였다.

Ⅱ. 선행 연구

CNN은 컴퓨터 비젼 분야에서 많은 연구가 활발히 진행 중인 인공신경망의 한 종류이며, 앞에서 설명한 것과 같이 일반적인 신경망과 매우 유사하다. CNN은 학습 가능한 가중치(weight)와 바이어스(bias)로 구성되어 있고, 일반적으로 Multi-Layer Neural Network와 유사하므로 몇 개의 층으로 이루어져 있다. 기본적인 CNN의 구조는 Convolution Layer, Pooling Layer, Fully-connected Layer의 3가지의 다른 층을 가지고 있고, 경우에 따라 여러 Layer가 추가되기도 한다.

CNN은 위에서 설명한 Convolution Layer와 Pooling Layer를 번갈아 사용하며 Convolution Feature를 추출하고, 최종적으로 추출된 Feature를 Fully-connected Layer를 이용하여 분류를 진행한다. 학습 과정에서 모든 Layer에 대한 추측을 진행할 때는 Forward Propagation, 각 Layer의 가중치와 바이어스 등의 파라미터를 갱신할 때는 Backpropagation 과정을 진행한다.

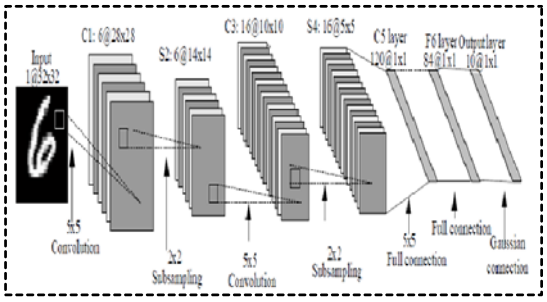

그림 1. MNIST 테스트를 위한 CNN의 구조[4]

위 <그림 1>은 MNIST Dataset을 인식하기 위한 CNN의 구조를 보여주며, C는 Convolution Layer, S는 Pooling Layer, F와 Output Layer는 Fully-connected Layer를 나타낸다.

그러나 위에서 설명한 것과 같이 CNN은 보통 하나의 영상에서 하나의 대응 값만을 추측하는데, 하나의 영상에서 여러 개의 물체를 검출하는 RCNN 중 성능이 좋은 것들은 Region Proposal 알고리즘에 의존적이다. 이 중 RCNN 류의 Detection Network의 수행시간을 감소시킨 SPPnet[5]이나 Fast-RCNN[6]은 성능은 좋지만 Region Proposal 알고리즘의 시간은 여전히 오래 걸리는 문제점이 존재했으며, 이로 인해 Region Proposal을 Neural Network로 Object의 경계와 각 위치에서의 Object Classification Score를 예측하는 Region Proposal Network(RPN) 이라는 것을 도입한 Faster-RCNN이 나타났다.

Faster-RCNN은 위에서 설명한 Region Proposal의 시간적인 문제를 RPN을 통해 해결한 알고리즘으로써, RCNN류 알고리즘의 Region Proposal에서 많이 사용되는 방법인 Selective-Search라는 Low-Level Feature에 기반한 Region들을 마구 통합하는 방식이 아닌 RPN을 사용한 방식이다. Fast-RCNN 같은 Region 기반의 Detector에서 사용되는 Convolution Feature Map은 Region Proposal에도 사용될 수 있으므로, Region Proposal 계산 시 소모되는 비용을 상당히 줄일 수 있었으며, Detector와 RPN을 결합하여 동시에 학습 시킬 수 있는 End-to-End 방식을 이용할 수 있게 된다. 또한 기존의 Selective-Search 방식을 사용한 Fast-RCNN보다 RPN을 이용한 Faster-RCNN이 실험결과 상 더 좋은 성능을 보였으며, 속도적인 측면에서 상당히 빠른 속도를 보여주었다. 그러므로 더욱 큰 영상이나 실시간 처리에 가까운 Object Detection 문제에서 좋은 성능과 빠른 속도로 수행할 수 있게 한다.

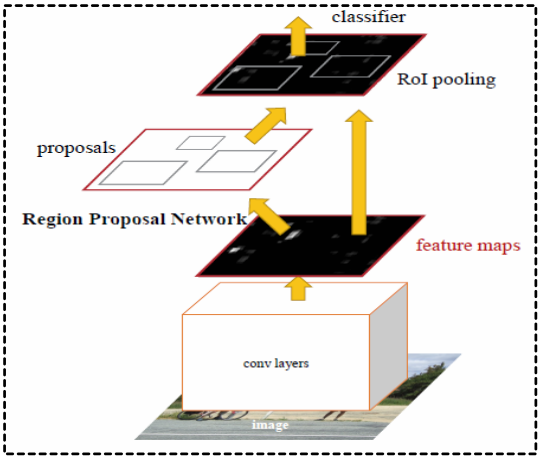

그림 2. Faster-RCNN Network의 구성[3]

<그림 2>는 Object Detection을 위한 Faster-RCNN의 전체 시스템을 보여주며 Convolution Layer를 공유하는 RPN과 이를 이용하는 Fast-RCNN Detector로 구성되어 있는 모습을 보여준다.

Ⅲ. Faster-RCNN을 이용한 PCB 부품 인식

본 논문에서는 위에서 설명한 Faster-RCNN을 이용하여 PCB에 실장 된 부품을 인식하는 방식을 진행한다. 먼저 카메라를 이용하여 해당 PCB의 영상을 촬영한다. 해상도는 3904x3904의 영상을

촬영하였고, 이 영상을 2600x2600 영상으로 영상의 크기를 Resize하는데, 이는 실험에 사용된 GPU의 한계로 원본 영상을 테스트할 수 없어 2600x2600으로 Resize 해서 사용했으며, 가장 작은 부품의 크기를 너무 작게 영상을 촬영하면 분류가 쉽지 않기 때문에 어느 정도 크기를 유지하도록 촬영하도록 한다. 이 후 촬영한 영상에서 실장 된 부품들의 Annotation 작업을 진행한다. Annotation 작업은 Training Set과 Test Set에서의 학습용 부품 데이터와 성능 측정에서의 사용을 위해 Annotation 작업을 진행하는 것이므로 정확하게 작업을 진행하며, 각 부품의 라벨 값과 Left, Top, Right, Bottom 데이터들을 저장하여 부품의 사각 영역을 저장한다. <그림 3>은 Resize한 영상에서 별도의 툴을 이용하여 Annotation된 부품들의 라벨과 부품의 영역을 표시한 영상이다.

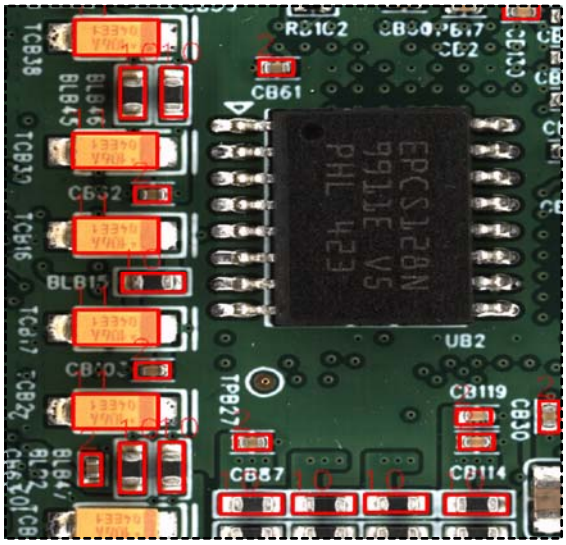

그림 3. PCB에 실장 된 부품들의 Annotation

본 논문에서는 학습 데이터의 수가 많지 않은 IC의 경우 Annotation 하지 않았으며, 영상 내의 실장 된 부품이 정확하게 나타나지 않은 데이터들 또한 포함시키지 않았다.

이 후 위에서 설명한 Annotation 된 데이터들과 원본 영상을 이용하여 Faster-RCNN의 RPN과 Detector를 학습시킨다. Network는 본 논문에서 RPN과 Detector에서 공유하는 Convolution Layer 층은 VGG16[7]의 D타입을 사용하며 13개의 공유 가능한 Convolution 층을 가진다. 이 후 Region Proposal을 생성하기 위해 마지막 공유층에서 만들어진 Feature를 이용하는 네트워크를 사용한다. 네트워크의 Dimension은 512 Dimension의 Feature로 차원이 줄어들며, 이 Feature는 Box-Regression층과 Box-Classification 층으로 나눠져 입력된다. <그림 4>는 위에서 설명한 내용을 그림으로 나타낸 것이다.

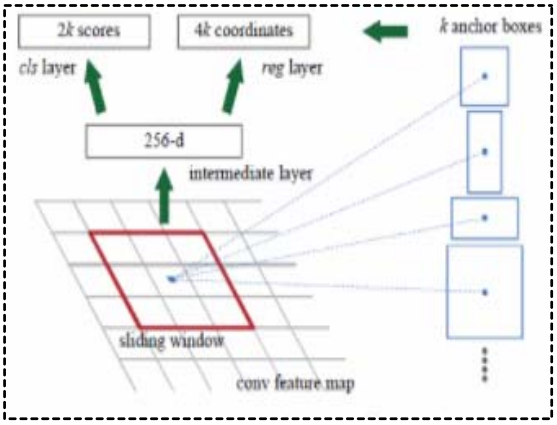

그림 4 . Faster-RCNN의 세부적 구조[3]

이 네트워크의 입력의 개수는 네트워크에 학습할 부품의 개수 + Background까지 1개를 추가한 개수가 입력의 수가 되고, 마지막 Box-Regression은 4 X (개수 + Background), Box-Classification은 개수 + Background가 된다. 이 후 학습은 전체 네트워크를 학습시키는 End-to-End 방식으로 학습을 진행하게 되고, 학습된 Weight와 Bias를 저장한 후 Test Set를 이용하여 테스트를 진행한다.

Ⅳ. 실험 결과

본 논문의 실험 환경은 Intel I7 6700K (up to 4.20 GHz, 4-Core), 16GB RAM, NVIDIA Geforce GTX 1070 Sli(8GB x 2)의 환경이며 운영체제는 Microsoft Windows 10 (64bit), 통합 개발 환경(IDE)는 Microsoft Visual Studio 2013, PyCharm을 사용하였으며, 관련 라이브러리로 OpenCV 2.4.13[7], Caffe Framework[8], 공개된 py-Faster-RCNN[9]을 수정하여 사용하였다. 실험은 Faster-RCNN을 이용하여 PCB에 실장된 부품들을 인식하는 것과 Template Matching을 이용한 방법 두 가지의 속도와 성능비교를 진행하였으며, 학습은 Faster-RCNN에서 Annotation 데이터를 학습할 때 사용하는 Stochastic Gradient Descent(SGD) 방식을 사용하였으며, PCB에 실장된 데이터가 상당히 많기 때문에 Region Proposal의 최대 개수는 1000개를 이용하였고, 기존에 Faster-RCNN에서 사용하는 Resize 파라미터를 사용하지 않고, 겹치는 Resion Proposal을 최대한 제거하기 위해 Non Maximum Suppression(NMS)을 사용하는데 NMS 임계값은 0.7로 고정하여 사용하며, Detection Threshold의 경우 Score가 0.8 이상인 데이터만을 사용하여 잘못된 Object의 위치를 측정하지 않도록 실험적인 값을 적용하였다. 또한 분류할 라벨의 개수는 23개의 부품 데이터로 구성하며, PCB 영상 중 Training Set의 개수는 32개, Test Set의 개수는 15개로 구성되며,

Annotation 된 부품영상의 총 개수는 아래 <표 1>과 같다.

표 1. Annotation 된 부품 영상의 라벨별 개수

Label	Training Set	Test Set
1	975	383
2	1626	700
3	26	12
4	115	48
5	12	3
6	23	6
7	9	5
8	18	10
9	20	7
10	283	111
11	288	87
12	6	3
13	16	9
14	69	31
15	10	6
16	48	18
17	18	8
18	47	21
19	76	28
20	52	22
21	7	2
22	11	4
23	10	7

Faster-RCNN의 학습은 미니 배치 당 영상 2장씩 사용하며, Annotation된 영상의 ROI는 128개를 사용했으며, Iteration 200,000번 학습을 진행했고, 학습 영상을 Flip하면서 학습을 진행했다.

템플릿 매칭은 라벨별로 2장의 템플릿을 두고 Non Maximum Suppression의 값은 Width, Height에 대해 25 이하의 차이가 나면 찾은 ROI를 제거하도록 진행하고, 템플릿 매칭 방법은 openCV의 SQDIFF 방식의 매칭을 진행하였으며 Diff Threshold는 0.1로 지정하였다. Faster-RCNN과 템플릿 매칭의 Test Set에 대한 수행시간과 Mean Average Precision(mAP)의 결과는 다음 <표 2>와 같다.

표 2. 수행시간과 mAP

	Faster-RCNN	템플릿 매칭
mAP	0.951	0.413
수행시간	0.73 s	4.54 s

Ⅴ. 결 론

<표 2>의 실험 결과에 따르면 템플릿 매칭 방식보다 Faster-RCNN을 사용한 PCB 부품 인식의 수행 결과가 더 좋게 나타났으며, 수행시간 또한 상당히 줄어든 것을 볼 수 있다. 템플릿 매칭에서의 문제점은 밝기가 다른 영상들이나 영상의 각도가 회전되어 있으면 Score가 상당히 떨어지는 문제점이 나타났으며, 부품이 있는 곳이 아니더라도 수식의 계산 값이 유사하면 부품으로 인식하는 경우가 상당히 많이 나타났다. 또한 매칭 할 템플릿의 개수가 늘어날수록 속도가 상당히 느려지는 문제점도 존재한다. 이러한 부분들에서는 Faster-RCNN을 이용한 PCB 부품 인식이 상당히 좋은 결과를 보여주었으나, 여전히 문제점은 존재하는데 IC칩이 몰려있는 부분에서 Object Detection의 결과가 떨어지는 부분들이 발생하고, Detection 된 ROI 데이터 중 영상의 가장자리에 존재하는 Annotation은 결과가 나오지 않는 경우가 존재한다. 또한 밝기의 차이가 큰 Object의 경우 찾아내지 못하는 경우가 여전히 발생하는 문제점이 발견되었다. 또한 상대적으로 크기가 큰 IC 칩의 경우 학습세트나 테스트세트의 개수가 많이 부족하여 데이터 추가가 필요하다. 위에서 설명한 문제점들을 개선하면 매우 정확하고, 수행시간도 빠른 알고리즘의 사용이 가능하여 실제 산업현장에서 활용할 수 있을 것으로 보인다.

참고문헌

[1] Y. LeCun, L. Bottou, Y. Bengio, and P.Haffner, “Gradient-Based Learning Applied to Document Recognition”, Proceedings of the IEEE, vol. 86, no. 11, pp. 2278-2324, Nov. 1998.

[2] S. Ren, K. He, R. Girshick, J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”, In Neural Information Processing Systems(NIPS), 2015

[3] Syazana-Itqan K, Syafeeza A.R, Saad N.M, “A MATLAB-based Convolutional Neural Network Approach for Face Recognition System”, Journal of Bioinformatics and Proteomics Review, vol. 2, 1-5, Feb, 2016.

[4] S. Ren, K. He, X. Zhang, J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition”, CoRR, abs/1406.4729v2, 2014.

[5] R. Girshick, “Fast R-CNN”, arXiv:1504.08083, 2015.

[6] K. Simon, A. Zisserman, “Very Deep Convolutional Networks for Large Scale Image Recognition”, arXiv:1409.1556v6, Apr. 2015.

[7] OpenCV, http://www.opencv.org

[8] Caffe framework, http://caffe.berkeleyvision.org

[9] py-faster-rcnn, https://github.com/rbgirshick/py-faster-rcnn