스크린 이미지 매칭을 위한 Faster D2-Net

전혜원*, 한성수**, 정창성*
*고려대학교 전기전자공학과
**강원대학교 자유전공학부
uclacarol@korea.ac.kr, sshan1@kangwon.ac.kr, csjeong@korea.ac.kr

Faster D2-Net for Screen Image Matching

Hye-Won Chun*, Seong-Soo Han**, Chang-Sung Jeong*
*Dept. of Electrical Engineering, Korea University
**Dept. of Division of Liberal Studies, Kangwon National University,University

요 약

스마트 기기와 애플리케이션의 테스트를 위해 빠르고 정확하게 스마트 기기 화면 상에서 테스트가 필요한 위치를 추출해야 한다. 필요한 위치를 추출할 때 스마트 기기 화면과 테스트할 수 있는 영역의 매칭 방식을 사용하는데 이를 위해 이미지의 변형이 발생해도 원하는 영역의 matching point 을 빠르고 정확하게 추출하는 feature matching 방식의 D2-Net 의 feature extraction 모델과 fitting algorithm 을 변경하였다.

1. 서론

다양한 기기들과 애플리케이션이 현재의 시장의 흐름에 맞춰서 기획과 출시까지 빠르게 진행되고 있다. 그 속도에 맞춰 해당 기기가 여러 애플리케이션에서, 또는 해당 애플리케이션이 여러 기기에서 제대로 실행되는지 테스트가 필요하다. 이 테스트 과정에 기기와 애플리케이션 관계 하나하나마다 테스트를 진행하는 것이 아니라 하나의 애플리케이션에 맞춰서 다양한 기기들을 한 번에 테스트할 수 있도록 해야 빠른 속도로 테스트를 진행할 수 있다.

해당 애플리케이션을 각종 스마트 기기와 테스트하기 위해서 테스터가 테스트하기 원하는 영역이 기기 화면의 어느 위치에 존재하는지 알아야 한다. 이를 위해서 스마트 기기의 화면과 테스트하기 원하는 영역의 매칭(matching)을 사용해 스마트 기기의 화면 상에서의 위치 좌표를 찾을 수 있다. 또 한 테스트가 목적이기 때문에 높은 정확도와 빠른 속도가 요구된다. 이를 위해서 테스트하기 원하는 영역의 이미지와 테스트할 이미지를 가지고 template matching 을 할 수 있지만 이 경우 이미지의 변형에 취약하다. 그 결과 정확도도 당연하게 떨어진다.

정확도를 위해서 template matching 보다 변형에 영향을 덜 받는 feature matching 방식인 D2-Net [1] 을 베이스 모델로 사용했다. 하지만 D2-Net 의 경우 한 이미지 당 1 초 이상의 시간이 걸리고 feature extraction 단계에서 가장 오래 걸린다.

이 논문에서는 feature extraction 속도를 줄이기 위해 기존 D2-Net 의 VGG16 을 스크린 이미지에 적합하게 변경한 VGG19 로 변경했다. 또 매칭 영역의 정확도를 위해 fitting algorithm 을 변경했다. 이를 통해 기존의 D2-Net 보다 빠르고 정확한 결과를 확인할 수 있었다.

2. 관련 연구

2.1 VGGNet

VGGNet[2]의 핵심은 모델이 깊게 만드는 것이 성능에 어떤 영향을 주는지 연구하고 깊이가 깊어질수록 성능이 좋아진다는 것을 확인했다. 실험을 위해 커널의 사이즈를 3x3 으로 설정했는데 그 이유는 모델을 깊게 만들기 위해서는 이미지의 크기를 천천히 축소되게 하기 위해서이다. 총 6 가지의 구조를 실험했는데 그중 VGG16 과 VGG19 의 구조의 차이점은 합성곱(convolution) 연산의 개수이고 VGG19 이 VGG16 보다 더 깊은 모델이다. 그 차이가 3 번째 layer 부터 존재하는데 VGG19 의 3 번째 layer 부터 합성곱의 수가 한 개 더 많아지고 총 3 개의 합성곱 연산이 추가되었다.

2.2 ResNet

ResNet[3]은 무조건 깊이가 깊어질수록 성능이 좋아지는지를 연구한다. 그렇지 않다는 실험 결과를 통해 Residual Block 을 제안한다. 기존 신경망의 residual 을 줄이기 위함이다. VGG19 의 구조를 가져가고 거기에 합성곱과 Residual Block 을 더해준다.

이를 통해 layer 수를 늘리고 특징 맵의 크기는 절반으로 줄이게 된다.

3. Faster D2-Net 구조

기존 D2-Net 의 경우 VGG16 을 사용해서 feature extraction 을 수행하게 되는데 그 결과를 가지고 detection 과 descriptor 를 동시에 얻게 된다. 이 과정은 기존의 연구들과 달리 detection 을 마지막 단계에 수행한다. 그 결과로 얻게 되는 keypoint 들이 더 안정적이므로 계절의 변화, 밤낮의 변화, 그림과의 매칭에서도 높은 정확도를 보여준다. 그래서 변형이 존재하는 스마트 기기 화면에도 적합하다고 판단했고 실제로 template matching 보다 높은 정확도를 얻을 수 있었다.

D2-Net 에서 사용하는 VGG16 는 training 와 testing 단계에 사용하는 모델의 구조가 조금 다르다. Training 단계에서는 VGG16 의 4 번째 layer 의 3 번째 합성곱까지 사용한다. Testing 단계에서도 VGG16 의 4 번째 layer 의 3 번째 합성곱까지 사용하지만 training 의 구조와는 다르게 4 번째 layer 의 3 개의 합성곱 연산에서 dilation 값을 2 로 설정했다.

Layer	Stride	Dilation	ReLU	Resolution
Input(256 x 256) - 3 channel.				x 1
Conv1_1 : 3 x 3, 64 channel.	1	1	O	x 1
Conv1_2 : 3 x 3, 64 channel.	1	1	O	x 1
Pool1 : 2 x 2, max.	2	1		x 1/2
Conv2_1 : 3 x 3, 128 channel.	1	1	O	x 1/2
Conv2_2 : 3 x 3, 128 channel.	1	1	O	x 1/2
Pool2 : 2 x 2, max.	2	1		x 1/4
Conv3_1 : 3 x 3, 256 channel.	1	1	O	x 1/4
Conv3_2 : 3 x 3, 256 channel.	1	1	O	x 1/4
Conv3_3 : 3 x 3, 256 channel.	1	1	O	x 1/4
Pool3 : 2 x 2, max.	2	1		x 1/8
Conv4_1 : 3 x 3, 512 channel.	1	1	O	x 1/8
Conv4_2 : 3 x 3, 512 channel.	1	1	O	x 1/8
Conv4_3 : 3 x 3, 512 channel.	1	1		x 1/8

(그림 1) 기존 D2-Net training 구조

Layer	Stride	Dilation	ReLU	Resolution
Input(~ 1200 x 1600) - 3 channel.				x 1
Conv1_1 : 3 x 3, 64 channel.	1	1	O	x 1
Conv1_2 : 3 x 3, 64 channel.	1	1	O	x 1
Pool1 : 2 x 2, max.	2	1		x 1/2
Conv2_1 : 3 x 3, 128 channel.	1	1	O	x 1/2
Conv2_2 : 3 x 3, 128 channel.	1	1	O	x 1/2
Pool2 : 2 x 2, max.	2	1		x 1/4
Conv3_1 : 3 x 3, 256 channel.	1	1	O	x 1/4
Conv3_2 : 3 x 3, 256 channel.	1	1	O	x 1/4
Conv3_3 : 3 x 3, 256 channel.	1	1	O	x 1/4
Pool3 : 2 x 2, avg.	1	1		x 1/4
Conv4_1 : 3 x 3, 512 channel.	1	2	O	x 1/4
Conv4_2 : 3 x 3, 512 channel.	1	2	O	x 1/4
Conv4_3 : 3 x 3, 512 channel.	1	2		x 1/4

(그림 2) 기존 D2-Net testing 구조

그림 1 은 기존 D2-Net 의 training 구조이고 그림 2 는 testing 구조이다.

하지만 D2-Net 는 이미지 한 장당 1 초 이상 걸리는데 그 이유는 VGG 계열이 ResNet 과 같은 다른 계열의 모델들 보다 연산하는 특징 맵(feature map)의 크기가 천천히 감소하기 때문이다. 특징 맵의 크기가 천천히 감소하기 때문에 더 많은 feature 를 얻을 수 있지만 그만큼 오래 걸리는 것이다. 하지만 속도를 줄이기 위해서 특징 맵의 감소를 빠르게 할 수 없는 이유는 특징 맵이 작아지고 모델이 깊어질수록 얻을 수 있는 feature 의 수가 한정돼서 정확도가 떨어지기 때문이다.

그래서 연산하는 특징 맵의 크기가 천천히 감소하되 높은 정확도와 빠른 속도로 detection 과 descriptor 를 추출하기 위한 feature extraction 모델의 변경이 필요했다.

속도와 정확도를 모두 안정적으로 구하기 위해 VGG16 대신 VGG19 를 선택해서 training 과 testing 을 위한 구조를 변경했다. 적절한 feature 를 추출하기 위해서 VGG19 에서 4 번째 layer 의 1 번째 합성곱까지의 feature 를 사용해서 기존 D2-Net 의 training 모델과 동일한 resolution 의 feature 를 사용했다.

Layer	Stride	Dilation	ReLU	Resolution
Input(256 x 256),3 channel.				x 1
Conv1_1 : 3 x 3, 64 channel.	1	1	O	x 1
Conv1_2 : 3 x 3, 64 channel.	1	1	O	x 1
Pool1 : 2 x 2, max.	2	1		x 1/2
Conv2_1 : 3 x 3, 128 channel.	1	1	O	x 1/2
Conv2_2 : 3 x 3, 128 channel.	1	1	O	x 1/2
Pool2 : 2 x 2, max.	2	1		x 1/4
Conv3_1 : 3 x 3, 256 channel.	1	1	O	x 1/4
Conv3_2 : 3 x 3, 256 channel.	1	1	O	x 1/4
Conv3_3 : 3 x 3, 256 channel.	1	1	O	x 1/4
Conv3_4 : 3 x 3, 256 channel.	1	1	O	x 1/4
Pool3 : 2 x 2, max.	2	1		x 1/8
Conv4_1 : 3 x 3, 256 channel.	1	1		X 1/8

(그림 3)Faster D2-Net 의 training 구조

그림 3 은 Faster D2-Net 의 training 구조이다. 이를 통해 training 단계에서도 기존보다 적은 채널 수와 합성곱의 개수로 빠른 속도로 많은 feature 를 얻을 수 있다.

Layer	Stride	Dilation	ReLU	Resolution
Input(256 x 256) - 3 channel.				x 1
Conv1_1 : 3 x 3, 64 channel.	1	1	O	x 1
Conv1_2 : 3 x 3, 64 channel.	1	1	O	x 1
Pool1 : 2 x 2, max.	2	1		x 1/2
Conv2_1 : 3 x 3, 128 channel.	1	1	O	x 1/2
Conv2_2 : 3 x 3, 128 channel.	1	2	O	x 1/2
Pool2 : 2 x 2, max.	2	1		x 1/4
Conv3_1 : 3 x 3, 256 channel.	1	2	O	x 1/4
Conv3_2 : 3 x 3, 256 channel.	1	2	O	x 1/4
Conv3_3 : 3 x 3, 256 channel.	1	2	O	x 1/4
Conv3_4 : 3 x 3, 256 channel.	1	2	O	x 1/4
Pool3 : 2 x 2, avg.	1	2		x 1/4

(그림 4) Faster D2-Net 의 testing 구조

그림 4 는 Faster D2-Net 의 testing 구조이다. 기존 testing 구조에서는 training 와 동일한 layer 를 사용했지만 변경된 testing 모델은 training 모델과 다르게 3 번째 layer 까지만 사용했다. 또 3 번째 layer 의 모든 연산에 dilation 값을 2 로 설정했다.

Dilation 은 Dilated Convolution 으로 kernel 사이의 간격을 더해줘서 receptive field 을 키우는 방법이다. 예를 들어 5x5 kernel 에 dilation 의 값을 2 로 주면 7x7 kernel 이 된다. 즉 25 개의 파라미터를 가지고 7x7 kernel 과 동일해진다. 이것을 사용할 경우 적은 파라미터로 receptive field 가 넓어지기 때문에 spatial dimension 의 손실이 줄어들고 빠르게 연산할 수 있게 된다.

Testing 단계에서 속도를 줄이고 공간적 손실을 줄이기 위해서 dilation 이 필요했고 D2-Net 보다 더 추가해서 3 번째 layer 에 모두 dilation 을 사용하게 되었다.

Keypoint 를 최종 결과로 얻어서 각 이미지의 point 들을 매칭 시키고 fitting algorithm 을 사용해 정확한 matching point 를 찾게 되는데 기존 D2-Net 의 경우 fitting algorithm 으로 RANSAC(RANdom Sample Consensus)[4]을 사용했다. 하지만 RANSAC 의 경우 정확한 keypoint 가 없고 어려운 모델의 경우 시간이 오래 걸리고 정확한 모델을 얻기 힘들다. 스마트 기기의 화면에서는 정확한 keypoint 로 모델링하기 어려워서 RANSAC 으로 얻을 수 있는 matching point 수가 매우 부족했다. 그래서 Faster D2-Net 에서는 RANSAC 대신 MAGSAC(Marginalizing Sample Consensue)[5]을 사용했다. MAGSAC 은 정확한 keypoint 가 많지 않은 경우에도 적은 반복으로 RANSAC 보다 빠르고 정확한 matching point 를 얻을 수 있기 때문에 스마트 기기 화면 상에서도 많은 수의 point 를 얻었다.

4. 실험 방법과 결과

Training. Faster D2-Net 을 training 하기 위해서 두 가지 dataset 을 사용했다.

첫 번째는 D2-Net 에서 사용한 Megadepth[6] 데이터 셋을 사용하였다. Megadepth 데이터 셋은 존재하는 구조물들을 다양한 각도와 위치에서 촬영해서 해당 구조물의 depth 값을 가지는 dataset 이다. 두 번째는 Faster D2-Net 의 목적인 스마트 기기 화면의 매칭을 위해 스마트 기기 화면을 dataset 으로 추가했다. 이때 데이터 셋으로 정의한 스마트 기기의 화면은 애플리케이션 상에서 동일한 기능을 하는 화면이고 애플리케이션의 데이터 셋 구성에 한계가 존재하기 때문에 임의의 변형을 주어서 스마트 기기 화면 데이터 셋을 생성하였다. 이렇게 두 가지 데이터 셋으로 Training 을 하였다.

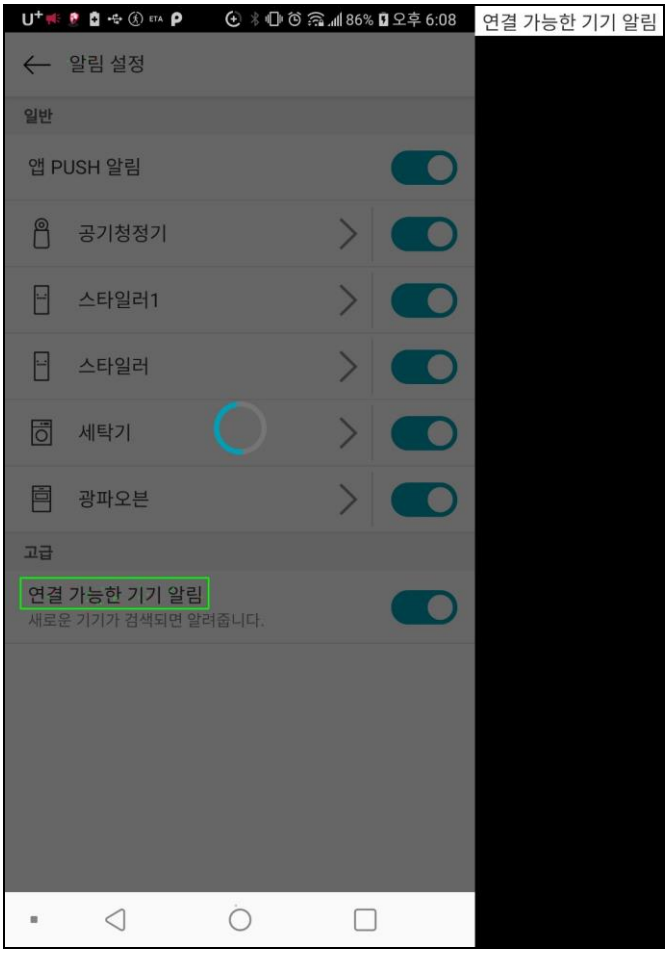
Evaluation. Faster D2-Net 의 성능 평가를 위해 스마트 기기 화면을 캡처한 데이터 셋을 사용하였다. 이 데이터 셋은 한 가지 애플리케이션을 3 가지 해상도 (1440x2880, 1440x2560, 1080x1920)의 기기에서 실행되는 화면을 캡처한 것이다. 이 데이터 셋을 구성하는 이미지에는 공백 존재, 색상과 크기 변형이 존재해서 Faster D2-Net 이 변형에 영향을 받는지 확인할 수 있다.

스마트 기기 화면에서 매칭된 영역을 표현하기 위해 첫 번째 영역을 기준으로 테스트하기 원하는 영역의 크기만큼 넓이로 정해 스마트 기기 화면에 표시해 주었다. 정확도 확인을 위해 표시된 영역의 동일 여부를 사용했다.

<표 1> D2-Net 과 Fast D2-Net 성능 비교

Mothed	Accuracy(%)	Speed(sec.)
D2-Net	89.30%	1.79
Faster D2-Net	90.20%	0.70

표 1 은 D2-Net 과 Faster D2-Net 의 최종 성능 비교 결과 비교이고 D2-Net 은 기존 D2-Net 을 수행한 것이다. 정확도를 비교해보면 기존 D2-Net 보다 1% 이상 올랐고 속도는 기존 D2-Net 은 1 초 이상의 걸리는 것에 비교해 Faster D2-Net 은 1 초 이상 감소한 0.7 초를 확인할 수 있었다.



(그림 5) Faster D2-Net 의 결과 이미지

그림 5 는 Faster D2-Net 의 최종 결과 이미지이다. 오른쪽 이미지는 테스트를 원하는 영역이고 왼쪽 이미지는 스마트 기기 화면 위에서 테스트를 원하는 영

역을 찾아 표시 해둔 것이다. 매칭하려는 두 이미지의 색상이 변형되어도 feature 를 찾아 기기 화면 위에서 테스트를 원하는 영역을 추출할 수 있다.

5. 결론

스마트 기기와 애플리케이션의 테스트를 할 때 테스트하는 영역을 탐색하기 위해서 기기 화면과 테스트하고 싶은 영역의 매칭 방식을 사용한다. 이때 정확하고 빠르게 기기 화면 상에서 test 하고 싶은 위치를 추출하기 위해서 faster D2-Net 을 제안한다. Faster D2-Net 은 매칭하려는 이미지의 변형에도 영향을 덜 받는 feature matching 기반의 D2-Net 의 속도와 정확도 높였다. 스마트 기기 화면에 적합한 feature 를 빠르게 추출하기 위해 feature extraction 모델을 변경하고 MAGSAC 을 사용해서 matching point 의 정확도를 올렸다. Faster D2-Net 을 통해 기존 D2-Net 보다 정확도는 1% 증가, 속도는 0.7 초 이하로 성능을 확인할 수 있었다.

이 연구는 스마트 기기와 애플리케이션을 테스트할 때 사용한 매칭 모델인 D2-Net 의 속도를 단축시키므로 스마트 기기를 사용하는 자동차 등의 장비에서도 실용적으로 사용될 수 있다.

향후 연구할 과제로는 현재 모델의 keypoint 들이 부족하고 sparse 하게 존재하기 때문에 MAGSAC 에 의존하게 된다. Keypoint 수를 늘리고 밀집되게 존재한다면 모델 자체의 성능을 올릴 수 있다.

이 논문은 2021 년도 4 단계 BK21 사업에 의하여 지원되었음.

참고문헌

[1] Dusmanu, Mihai and Rocco, Ignacio and Pajdla, Tomas and Pollefeys, Marc and Sivic, Josef and Torii, Akihiko and Sattler, Torsten.: “D2-Net: A Trainable CNN for Joint Detection and Description of Local Features.” In: Processdings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp.8092-8101

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Proc. ICLR, 2015

[3] HE, Kaiming, et al. “Deep residual learning for image recognition.” In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. pp. 770-778.

[4] M. Fischler and R. Bolles, “Random sample consunsus: A paradigm for model fitting applications to image analysis and automated cartography,” Proc. Image Understanding Workshop, Apr. 1980, pp. 71-88

[5] Barath, Daniel and Matas, Jiri and Noskova, Jana.: “Magsac : marginalizing sample consensue.”. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019, pp.10197-10205

[6] LI, Zhengqi; SNAVELY, Noah. “Megadepth: Learning single-view depth prediction from internet photos.” In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018. pp. 2041-2050.