모바일 환경 영상인식을 위한 신경망기반 Speeded Up Robust Features 차원 감소

윤두밈○ 김경중1)
세종대학교 컴퓨터공학과
krad@hanmir.com, kimkj@sejong.ac.kr

Dimensionality Reduction of Speeded Up Robust Features Using Neural Networks for Object Recognition in Mobile Environments

Du-mim Yoon○ Kyung-Joong Kim
Dept of Computer Engineering, Sejong University

요 약

최근에 스마트폰이 발달하고 대부분의 모바일 기기에 카메라가 달리면서 카메라를 이용한 애플리케이션 또한 늘어나고 있는데 기존의 PC상에서 로고 인식등을 위해 사용되는 SURF를 이용한 이미지 매칭에는 유클리드 거리 계산을 사용하고 있다. 그러나 이 방법으로는 PC보다는 사양이 낮은 모바일 기기에 적용하기에는 기존에 사용하고 있는 방법이 인식할 이미지마다 모든 특징점을 비교하는 방법을 사용하기 때문에 연산량이 높은 편이다. 본 논문에서는 미리 인식할 이미지를 뉴럴넷에 학습시킨 뒤, 뉴럴넷을 필터링으로 사용하여 일부의 특징점만을 비교해 연산량을 줄여서 속도를 향상시키는 방법을 제안하였으며 이를 이용하여 대략 30%가량의 성능 향상이 나타난 것을 알 수 있었다.

1. 서 론

최근에는 스마트폰, 랩탑부터 로봇까지 카메라를 기본적으로 장착하고 있는 장비가 많아지고 있으며 또 이와 맞물려 카메라를 이용한 이미지의 프로세싱 등에 대한 관심 또한 더욱 더 높아지고 있는데, 그 중 하나가 물체인식이다. 물체인식이란 카메라로 바라보고 있는 이미지가 무엇인가 인식하는 것인데, 사람의 경우 손쉽게 그것을 해내는 반면 컴퓨터는 그렇지 않아 아직까지도 수많은 연구가 이루어지고 있는 분야이다.

이런 물체인식을 위해 사용되는 것으로는 Scale-Invariant Feature Transform[1]가 있는데, 이 방법은 grayscale의 이미지상에서 크기나 회전에 불변하는 특징점을 사용하는 것으로 이 경우 성능은 좋은 편이나 매우 긴 연산시간을 갖고 있다는 결점이 있다. 따라서 SIFT를 바탕으로 속도를 향상시킨 Speeded-Up Robust Features[2]를 주로 사용되게 되는데 이 두 경우 모두 유클리드 기반의 거리 비교로 이미지 검출 및 인식을 수행하게 된다.

한 이미지를 SURF로 추출시킨 결과는 x좌표, y좌표, 사이즈, 라플라시안값, 128차원의 벡터로 이루어진 N개의 이미지 특징들로 나타나게 된다 (그림 1). 이 특징점들을 비교할 때, 두 특징점을 가지고 Euclidean Distance를 이용하여 라플라시안값과 128차원 벡터를 이용해 거리계산을 한뒤 임계값보다 낮을 경우 같은 특징점으로 판단을 하는 방법을 쓰는데 이 경우 특징점의 수와 연산량이 정비례하게 된다.

따라서 기존에 사용된 방법으로는 이미지의 특징점을 찾는 속도를 향상시키기 위해 SIFT에 PCA를 적용한 PCA-SIFT[3]나 SURF의 알고리즘을 병렬처리한 P-SURF[4] 등이 있었지만, PCA-SIFT는 SURF에 비해 그리 빠르지 않고 P-SURF는 연산을 병렬처리한 것일 뿐, 연산량 자체를 감소시키지 못했다. 또 Saliency map을 이용해 속도를 향상시킨 방법[5]도 있었지만, 이 방법의 경우 이미지 매칭에만 사용되었고, 특징점의 개수를 줄여 비교 횟수를 줄이는 방법이다.

이와 비교해 본 논문에서는 뉴럴넷[6]을 이용해 특징점 비교시에 나타나는 연산량을 감소시키는 방법을 사용하였다.

먼저 인식할 필요가 있는 이미지의 특징점과 인식할 필요가 없는 이미지의 특징점을 학습시킨 뒤, 입력 이미지를 SURF를 통해 특징점을 얻어낸다. 그 다음 얻어낸 특징점을 뉴럴넷을 이용하여 이 특징점이 인식할 필요없는 특징점인지, 혹은 인식할 필요가 있는 이미지라면 어떤 이미지의 특징점인지 구별하여 아니라고 판단되면 Reject시키고[7] 맞다고 판단되면 Euclidean Distance연산을 수행시킨다.

1) 교신저자

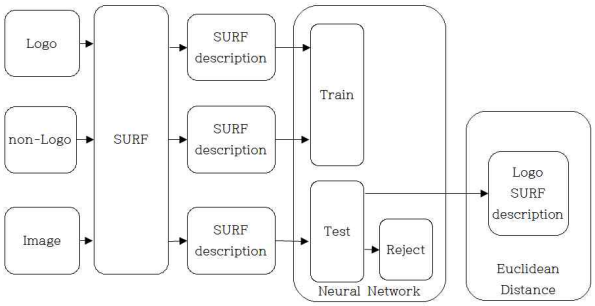

그림 1 제안하는 방법의 블록 다이어그램

이와 같은 알고리즘을 사용하게 되면, 로고의 개수와 연산량이 비례관계가 되지 않기 때문에 모든 특징점마다 거리를 계산하는 것보다 연산량이 적어질 수 있다. 따라서 실제 연산량을 측정하기 위해 SURF를 통해 특징점을 추출한 뒤 모든 특징점에 대해 거리만으로 계산했을 때와 신경망을 거쳐 거리를 측정했을 때의 시간을 구해 비교해보기로 한다.

2. Neural Network 필터링을 이용한 SURF 성능 향상 기법

본 논문에서는 SURF(SURF-128)를 통해서 얻은 feature point descriptor를 가지고 매칭할 때에 신경망을 써서 관계 없다고 판단되는 feature를 reject시킴으로써 매칭의 속도를 증가시킨다.

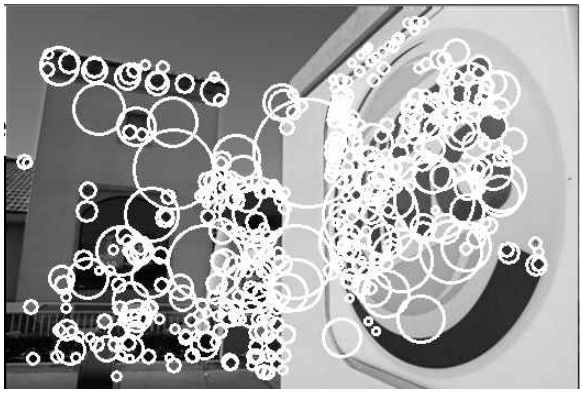

그림 2 SURF를 통해 얻어낸 특징점들

이 신경망은 128개의 입력 레이어와 64개의 히든 레이어, 학습시킬 로고 개수만큼의 출력 레이어로 총 3개의 레이어로 이루어져 있다. activation function은 0.0~1.0 출력값의 sigmoid 함수이다.

$$sigmoid(input) = \frac{1}{1+e^{-input}}$$

입력 뉴런에는 노멀라이즈된 128개의 벡터값이 들어가고 출력 뉴런에는 각 로고의 판단 값이 나온다. 이를 위해 먼저 신경망을 학습시키는데 초기의 weight값은 렌덤으로 주었고 각각의 로고 이미지는 해당 로고의 출력값만 1로 학습시키고 로고가 아닌 이미지는 모든 출력 뉴런을 0으로 학습시킨다. 학습시킬 때는 로고와 비로고의 비율을 비슷하게 맞춰 BackPropagation[8]을 이용해 weight값을 조절한다. 또 에러율이 일정 수준 미만(여기서는 1%)으로 떨어질 때까지 학습을 시키는데 트레이닝 샘플이 몰려 있으면 전체적으로 학습이 이루어지지 않은 상태에서 에러율이 낮아질 수 있기 때문에 일정 epoch 후 트레이닝 셋을 섞어준다.

이렇게 신경망의 트레이닝이 끝났다면 테스트에 사용한다. 먼저 로고와 입력 이미지에 대해 SURF로 feature point들을 얻어낸 뒤에 각각의 입력 이미지의 feature point로 입력 뉴런에 128개의 노멀라이즈된 벡터 값을 입력하고 신경망에 일정값 이상(여기서는 0.9)의 출력 뉴런 값이 나타난다면 그 출력 뉴런의 로고로 판단하고 해당 로고의 전 feature point와 Euclidea Distance로 거리를 계산하여 일정값 미만(여기서는 0.3)이면 동일한 feature point라고 판단한다.

$$ED(F_{input}, F_{i=0\ to\ n\ in\ Logos}) = \sqrt{\sum_{n=0}^{127}(F_{input,n} - F_{i,n})^2}$$

그림 3 입력 이미지의 특징점들을 뉴럴넷으로 인식시킨 후의 분포도 (검은색은 Reject됨)

3. 실험 및 결과분석

본 논문의 실험 학습 데이터로는 인식할 이미지인 2개의 로고를 기본 형태와 각각 -45°,+45° 로 기울인 변형 이미지, 여백만을 증가시킨 뒤 가우시간 블러를 3만큼 주고 -45°,+45° 씩 기울인 변형 이미지와 함께 Reject용 이미지로 Google을 이용하여 해당 로고로 검색시 로고가 포함되지 않은 이미지를 로고당 20개씩 도합 2 * (6 + 20) = 52개의 이미지를 선택했다.

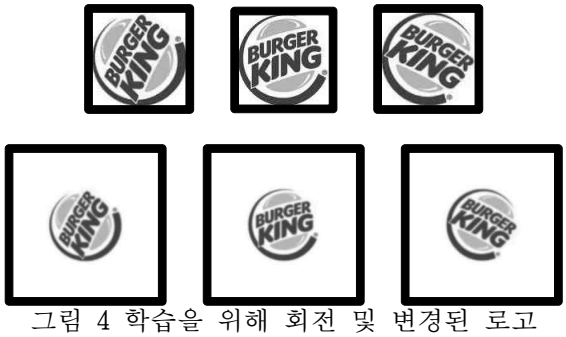

그림 4 학습을 위해 회전 및 변경된 로고

그리고 선택된 52개의 이미지를 SURF를 통해 이미지 각각의 특징점들을 추출하여 한 특징점의 128차원의 벡

터값과 해당 학습 출력값을 추가하여 하나의 샘플로 만든다. 이런 식으로 1631개의 샘플 셋을 만들어 학습시켰다.

Out layer	logo1	logo2	Reject
첫번째 값	1.0	0.0	0.0
두번째 값	0.0	1.0	0.0

표 1 뉴럴넷의 출력값 설정

학습 방법으로는 학습률을 0.1로 설정하고 노드의 활동함수로 시그모이드 함수를 적용하여 0.0~1.0의 출력이 나타나게 하였다.

또 에러율이 1%미만이 될 때까지 학습을 시켰고, 매 epoch마다 Backpropagation[8]을 이용해 weight를 변경시켰으며, 이로 인해 동일 샘플이 몰려 있을 때나 로고의 특징점 샘플 수에 비해 Reject의 샘플 수의 차가 심할 경우 에러율이 낮아진다 하더라도 실질적인 총 에러율은 별로 낮아지지 않는 현상이 나타나 이와 같은 현상을 막기 위해 2000 epoch당 한번씩 샘플셋을 샘플의 수만큼 랜덤으로 섞어주었다.

그리고 테스트 이미지로 Google에 해당 로고를 검색시 로고가 포함되어 있는 이미지를 선발했다.

테스트 이미지로 로고당 5개씩 총 10개를 뽑았고, 실험을 위해 모바일 환경에 앞서 약 2Ghz의 윈도우 기반의 PC상에서 알고리즘을 구현하였고, 시간 계산은 특징점 추출 후부터 뉴럴넷 필터링을 통해(필터 사용시) 매칭완료까지의 시간을 측정하였다.

	폭	높이	Logo 구분
이미지01	1229	922	1
이미지02	445	408	1
이미지03	700	978	1
이미지04	520	349	1
이미지05	328	249	1
이미지06	360	480	2
이미지07	564	376	2
이미지08	333	500	2
이미지09	640	480	2
이미지10	144	193	2

표 2 테스트에 사용된 이미지의 크기

위의 이미지에 대해 Euclidean distance의 Threshold 값은 0.3으로 주고 뉴럴넷을 사용했을 때와 안 사용했을 때의 특징점 인식 개수의 비교 결과는 다음과 같다. (logo1, logo2는 Euclidean distance 매칭으로 인식된 값의 개수이다)

	non-NN			NN-used		
Src Image	logo1	logo2	time (msec)	logo1	logo2	time (msec)
01	0	5	66	0	1	41
02	0	19	37	0	11	23
03	0	19	55	0	9	38
04	0	2	12	0	2	8
05	0	12	11	0	8	8
06	4	1	6	2	0	4
07	1	0	14	0	0	10
08	1	0	9	1	0	5
09	0	1	14	0	0	9
10	1	0	3	0	0	2

표 3 <non-NN 과 NN-used 사용시의 시간 소모>

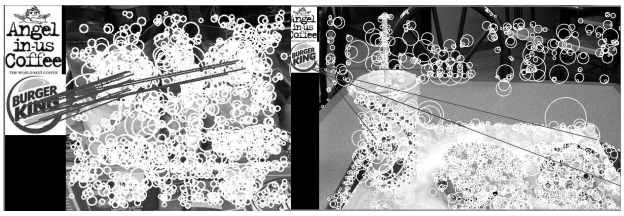

그림 5 non-NN의 결과, 특징점과 매칭된 선.

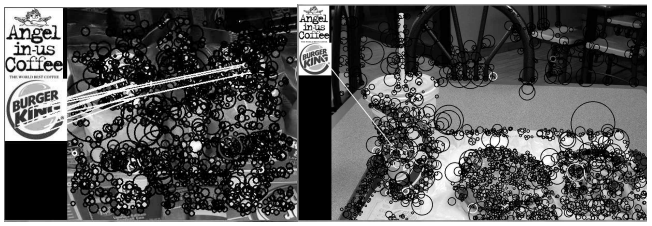

그림 6 NN의 결과, 다량의 Reject(검은색 원)와 소량의 인식점(어두운 원-1번로고/하얀 원-2번로고).

다음은 일치도를 비교해보기로 한다. 본 논문에서는 정확도를 확인하기 위해 로고 자체의 검출 여부를 비교하기보다는 원본 로고상에서 특징점의 위치가 매칭된 이미지의 특징점 위치가 동일한지를 비교하여 동일하다면 일치, 로고가 아닌 특징점과 매칭되거나 로고안에 매칭되었지만 원래 로고의 위치가 다를 경우 오류로 판단하기로 한다

이를 통해 위의 결과를 분석해보면 다음과 같다.

	non-NN			NN-used		
Src	매칭	일치	오류	매칭	일치	오류
01	5	1	4	1	1	0
02	19	19	0	11	11	0
03	19	19	0	9	9	0
04	2	2	0	2	2	0
05	12	11	1	8	8	0
06	5	4	1	2	2	0
07	1	1	0	0	0	0
08	1	1	0	1	1	0
09	1	0	1	0	0	0
10	1	1	0	0	0	0

표 4 <non-NN 과 NN 사용시의 일치도>

위의 표를 보면 알 수 있듯이 유사한 일치도를 보여주고 있으나 뉴럴넷을 사용하지 않은 경우에는 평균 79.2%, 사용한 경우에는 평균 70%로 뉴럴넷을 사용하지 않은 경우의 일치도가 높았지만, 오류율 역시 뉴럴넷을 사용했을 때보다 높았다.

다음으로는 연산 시간의 증가량을 비교해 보기로 한다.

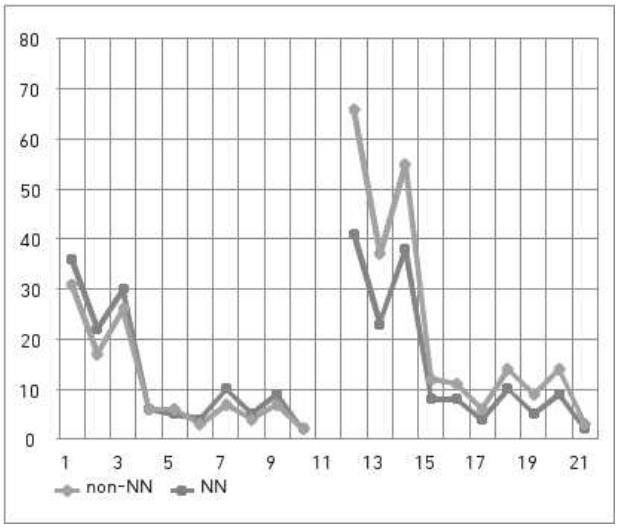

그림 6 1개의 로고만을 이용한 경우(좌) 와 2개의 로고를 이용해 비교한 경우(우)의 시간소모(msec)

1개의 로고만을 이용해 비교한 경우, NN은 non-NN의 연산시간과 비슷하거나 약 12%정도 더 느린 연산시간을 보였지만, 2개의 로고를 이용해 비교한 경우에는 정반대로 non-NN의 경우, 로고의 수만큼 알고리즘을 반복 사용해야 되기 때문에 1개의 로고를 이용할 때보다 약 2배 정도의 시간이 소모되는 반면, NN의 경우 뉴럴넷이 시간 필터링 역할을 해 non-NN에 비해 약 35%정도의 연산시간을 감소시킬 수 있었다.

또 Threshold의 값을 0.3에서 0.4로 증가시켜 테스트를 한 결과 연산시간은 비슷했으나 non-NN의 경우 비교하기 힘들 정도로 오류가 많아졌고 NN의 경우는 다음의 표와 같이 오류와 일치가 같이 증가했음을 볼 수 있었다.

4. 결론 및 향후연구

기존의 SURF의 매칭시에 소모되는 시간과 로고의 개수가 비례하는 반면에 뉴럴넷을 이용한 매칭에서는 비교할 로고의 수가 2개가 되어도 1개였을 때와 그다지 시간차이가 나지 않는 것을 볼 수 있었다. 또 학습을 시키기 때문에 학습시킨 로고 밖만을 매칭시킬 수 있었지만, 이로 인해 나타나는 결과로 유사하다고 잘못 판단될 특징점이 제거되어 정확도 또한 향상된 것을 알 수 있었다.

앞으로는 본 연구에서 수행한 것보다 더 많은 로고의 개수를 적용하여 실험할 것이고, 특징점 매칭이 아닌 다른 부분에서의 연산량 감소도 적용해 전체적인 SURF 매칭의 시간 감소를 강구할 계획이다.

감사의 글

이 논문은 2010년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업 (2010-0012876) 및 뇌과학 원천기술개발사업임 (2010-0018948).

참고문헌

[1] Lowe, D.G. “Object recognition from local scale-invariant features,” The Proceedings of the Seventh IEEE International Conference on Computer Vision, vol. 2, pp. 1150 - 1157, (1999).

[2] H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: Speeded up robust features,” in Proceedings of the European Conference on Computer Vision, pp. 404-417, (2006).

[3] O. G. Luo Juan. “A Comparison of SIFT, PCA-SIFT and SURF,” International Journal of Image Processing (IJIP) vol. 3, issue. 4, pp. 143-152, (2009).

[4] Zhang, N., “Computing optimised parallel speeded-up robust features (P-SURF) on multi-core processors,“ International Journal of Parallel Programming vol. 38, no. 2, pp. 138-158, (2010).

[5] Pimenov, V., “Fast image matching with visual attention and surf descriptors,“ Proceedings of19th International Conference on Computer Graphics and Vision, GraphiCon’ Moscow, Russia. pp. 49 -56. (2009).

[6] Aleksander, I. and Morton, H., An Introduction to Neural Computing 2nd edition. Chapman and Hall (1989).

[7] Le Dung, Mizukawa, M. “Designing a pattern recognition neural network with a reject output and many sets of weights and biases,” Pattern Recognition Techniques, Technology and Applications pp. 507-522 (2008).

[8] Russell, S. Norvig, P., Artificial Intelligence A Modern Approach. p. 578. (1995).