심층 네트워크의 과계산 문제에 대한 고찰

박다솔[1O], 손정우[2$], 김선중[2#], 차정원[1*]

창원대학교[1], 한국전자통신연구원[2]

{dasol_p [O],jcha[*]}@changwon.ac.kr, {jwson[$],kimsj[#]}@etri.re.kr

A study on the Problems of Overcomputation in Deep Networks

Da-Sol Park[1O], Jeong-Woo Son[2$], Sun-Joong Kim[2#], Jeong-Won Cha[1*]

Changwon National University[1], Electronic Telecommunications Research Institute[2]

요 약

딥러닝은 자연어처리, 이미지 처리, 음성인식 등에서 우수한 성능을 보이고 있다. 그렇지만 복잡한 인공신경망 내부에서 어떠한 동작이 일어나는지 검증하지 못하고 있다. 본 논문에서는 비디오 캡셔닝 분야에서 인공신경망 내부에서 어떠한 동작이 이루어지는지 검사한다. 이를 위해서 우리는 각 단계에서 출력층을 추가하였다. 출력된 결과를 검토하여 인공 신경망의 정상동작 여부를 검증한다. 우리는 한국어 MSR-VTT에 적용하여 우리의 방법을 평가하였다. 이러한 방법을 통해 인공 신경망의 동작을 이해하는데 도움을 줄 수 있을 것으로 기대된다.

주제어: 비디오 캡션, 셀프 어텐션, 기계 학습, 딥러닝

1. 서론

비디오 캡션 자동 생성 작업에 대한 관심이 점점 높아지고 있다. 예를 들어 YouTube에서는 1 분마다 수백 시간 분량의 비디오 콘텐츠가 업로드된다. 사람이 이러한 엄청난 양의 비디오를 해결할 수 없으므로 이를 검색하고 이해하는 새로운 기술이 많이 요구된다.

비디오 캡션 생성은 비디오에 대한 자막(즉, 자연어로 비디오를 설명하는 짧은 문장)을 생성하는 작업으로 이 문제를 해결하는 중요한 기술이며 시각 장애인을 위한 접근성을 크게 향상시킬 수 있다. 비디오 캡션에 대한 연구는 오래전부터 시작되었지만 비디오를 해석하는 모듈과 자연어로 생성하는 모듈에 대한 상호 작용이 필요하기 때문에 여전히 어려운 과제이다. 비디오를 이해하고 해석하기 위해서는 비디오 프레임의 흐름을 이해하고 연속된 프레임 사이 관계의 특징을 파악하는 능력을 가지고 적절한 특징을 추출할 수 있어야 한다.

비디오 캡셔닝은 일반적으로 비디오를 n개의 프레임으로 변환하고 특징을 추출한 후 추출한 특징을 이용하여 문장을 생성한다. 이 때, 생성된 출력은 문법적으로 정확한 단어 시퀀스가 되어야 한다.

비디오 캡셔닝 분야에서 딥러닝을 이용하는 연구가 많이 되고 있다. 딥러닝에서 네트워크의 깊이는 성능 향상에 영향을 미친다. 하지만 복잡한 인공신경망 내부에서 어떠한 동작이 일어나는지는 검증을 못하고 있으며, 어떠한 단계에서 학습이 올바른 방향으로 진행되고 있는지 확인하는 것이 중요해지고 있다.

본 논문은 비디오 캡션 생성 모델의 결과를 이용하여 학습의 효과를 확인하는 것이 아니라 네트워크 중간 단계의 캡션 생성기를 이용하여 캡션을 생성한다. 이를 통해 모델의 학습이 정상적으로 진행되고 있는지 확인하고 단계별 분석을 진행한다.

2. 관련 연구

학습 내 중간의 결과물을 추출하여 네트워크에 중간 결과를 출력하는 연구가 진행되었다. [1]은 문제를 해결하기 위해 필요 이상으로 생각하는 것을 '과하게 생각하는 것(overthinking)' 이라고 정의했으며 이는 딥 뉴럴 네트워크(Deep Neural Network)에서도 적용된다고 판단하였다. 이에 대한 근거는 최근 4개의 콘볼루션 뉴럴 네트워크(Convolution Neural Network) 모델과 이미지 분류 태스크에 대한 실험 진행 시 샘플의 계산량 낭비와 오분류 발생 확률이 높다는 것을 확인했다. 이러한 overthinking에 대한 탐지하는 방법은 내부 분류기를 사용하는 것이다. 내부 분류기는 각 콘볼루션 레이어에서 완전 연결 계층(Fully connected layer)을 통해 내부 예측의 결과를 내는 방식으로 학습 중간 단계의 결과물을 출력하여 각 단계별로 올바르게 분류하는지에 대한 연구를 진행하였다.

3. 제안 방법

그림 1은 본 논문에서 제안한 시스템을 보여준다. 중간 단계 캡션 생성기를 통해 n개의 캡션($O_1 \dots O_n$)을 생성하고, 이를 이용하여 분석을 진행한다. 콘볼루션 및 어텐션 등과 같은 레이어로 깊이를 깊게 만든 모델에 중간 단계 캡션 생성기를 추가함으로써 블랙박스라 표현되는 모델 중간 단계의 학습 경향을 간접적으로 확인할 수 있다. 그리고 분석 결과는 학습 구조의 방향을 잡을 수 있도록 도움을 줄 것이라고 예상된다.

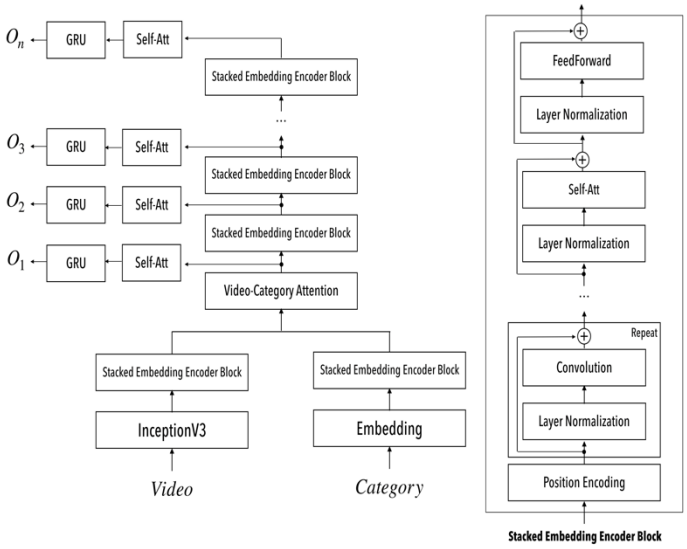

그림 1. 제안 모델의 구조

비디오 임베딩은 각 비디오의 프레임 중 100개를 임의로 추출하여 InceptionV3[2]의 결과를 사용한다. 카테고리 임베딩은 [3]과 동일하게 적용하였으며 단어 임베딩과 문자 임베딩의 조합을 사용한다. 단어 임베딩은 300차원의 학습된 fastText[4]를 이용하고 학습이 되지 않도록 설정한다. 문자 임베딩도 200차원의 학습된 fastText를 이용하고 학습이 되도록 설정하여 콘볼루션과 하이웨이(Highway) 네트워크를 통해 최종 문자 임베딩을 생성한다. 이 때 하나의 단어가 가질 수 있는 최대 문자의 수를 6으로 설정한다. 최종적으로 단어 임베딩과 문자 임베딩을 연결하여 사용한다.

스택 임베딩 인코더 블록은 콘볼루션이나 순환(Recurrent)을 사용하지 않아 위치 정보가 포함되지 않는다. 위치 정보를 이용하기 위해 삼각 함수 중 sine($2n$), cosine($2n-1$)을 이용하여 프레임 또는 단어의 위치정보를 추가하는 포지션 인코딩(Position encoding)을 진행한 후 레이어 정규화(Layer normalization)를 거친다.

분리 가능한 깊이별 콘볼루션 층(Depthwise separable convolution layer)[5,6]은 채널마다 독립적으로 콘볼루션을 실행하는 깊이별 콘볼루션(Depthwise convolution)과 1D CNN을 이용하여 하나의 새로운 채널로 합치는 위치별 콘볼루션(Pointwise convolution)을 설정한 각 층 수만큼 반복한다.

셀프 어텐션[7]은 각 입력으로 들어오는 비디오와 범주 정보를 전체 차원에 대해 설정한 헤드(head) 수 만큼 나누어 어텐션을 적용한다. 이 때 어텐션은 자기 자신을 잘 표현할 수 있는 비디오와 범주 정보 쌍을 찾아 이를 이용하여 결과를 생성한다. 그 이후 각 헤드가 생성해낸 셀프 어텐션이 치우치지 않도록 균등하게 섞는 역할을 하는 피드-포워드 층(Feed-Forward layer)을 거친다.

비디오-범주 정보 어텐션은 총 4개의 벡터를 연결하여 사용한다. 첫 번째는 인코딩된 비디오(V)와 범주 정보(G)를 이용해 계산된 유사도 매트릭스($S \epsilon R^{n \times m}$)이다. 두 번째는 계산된 유사도 매트릭스를 이용하여 각 행과 열에 소프트맥스(softmax)를 통하여 정규화된 매트릭스($\bar{S}, \bar{\bar{S}}$)이고 세 번째는 정규화된 매트릭스와 범주 정보를 이용하여 계산된 어텐션($A \in \bar{S} \times G^T \in R^{n \times d}$)이다. 이 때 사용되는 유사도 계산 함수는 [8]에서 제안되었고, $f(G,v) = W_0[G, v, G \odot v]$를 사용한다. 여기서의 $\odot$는 요소별 곱(element-wise product) 연산이고, W_0은 학습해야하는 가중치이다. 네 번째는 비디오와 범주 정보의 연관성을 찾기 위해 [9]에서 제안한 DCN인 $B = \bar{S} \cdot \bar{\bar{S}} \cdot G^T$를 적용한 비디오-범주 정보 어텐션($B$)이다. 최종적으로 인코더 벡터는 $[g, a, g \odot a, g \odot b]$를 사용하고, a와 b는 어텐션 매트릭스인 A와 B의 행을 의미한다. 최종 인코더 벡터는 스택 임베딩 인코더 블록을 3번 반복하여 나온 출력 벡터에 셀프 어텐션을 거친 벡터를 사용한다. 이 때, 스택 인베딩 인코더 블록과 셀프 어테션의 가중치는 공유된다.

인코더의 결과물은 비디오와 범주 정보에서 유익한 자질들을 추출하여 생성한 결과 벡터이며 디코더의 초기 상태로 주어진다. 디코더는 1-Layer GRU(Gated Recurrent Unit)을 사용하여 비디오에 대한 캡션을 생성한다. 디코더의 입력인 인코더의 결과 벡터와 GRU의 은닉 상태(Hidden state)의 차원은 256차원이다.

우리는 학습 도중 각 단계별로 문장을 올바르게 생성하는지에 대한 결과를 확인하기 위해 5개의 중간 단계 캡션 생성기를 사용한다. 중간 단계 캡션 생성기의 결과를 $O_i (0 \le i \le n)$이라고 명명하며, i는 스택 임베딩 인코더 블록의 반복 횟수를 의미한다. O_0은 스택 임베딩 인코더 블록을 0번 반복(비디오-범주 정보 어텐션의 셀프 어텐션 벡터)한 후 생성된 캡션을 의미하고, O_1은 스택 임베딩 인코더 블록을 1번 반복한 벡터로부터 생성된 캡션을 의미한다. 스택 임베딩 인코더 블록 반복 횟수 설정에 따른 마지막 모듈에 생성된 캡션이 제안 모델의 최종 결과 캡션이고 각 캡션을 생성하는 디코더는 동일한 디코더를 사용하고, 마지막 모듈이 생성된 캡션만을 이용하여 손실 함수를 계산한다.

본 논문에서 사용된 데이터는 MSR-VTT[10] 데이터셋으로 진행한 실험이고, 사용된 데이터의 전처리 및 통계는 [3]과 동일하게 적용하였다.

4. 실험 및 분석

우리는 총 4번의 실험을 진행했다. 기준이 되는 실험을 임베딩 인코더 블록을 4번 반복한 구조로 설정하였으며, 이 실험을 '모델 1'이라고 명명한다. 표 1은 기준 실험의 각 중간 결과별 성능을 측정한 결과를 나타낸다. B@1은 BLEU 1을 의미하고, R은 ROUGE_L, C는 CIDEr를 의미한다.

표 1 모델 1의 성능 결과표.

분류	B@1	B@2	B@3	B@4	R	C
O_0	53.8	42.9	35.4	29.5	42.2	23.4
O_1	72.9	59.7	50.6	43.1	59.3	**41.6**

O_2	**73.6**	**60.5**	**51.3**	**43.8**	**59.7**	39.8
O_3	72.1	59.3	50.1	42.6	58.2	40.0
O_4	66.0	52.9	43.9	36.7	52.7	32.9

표 1의 결과로 스택 임베딩 인코더 블록을 반복할수록 성능이 하락하는 현상을 볼 수 있었고 우리는 이를 overthinking 문제의 발생이라고 판단했다. 이것은 네트워크의 연산이 필요 이상으로 수행되어 오히려 노이즈로 작용한 결과이다. 추가적인 연산을 제거함으로써 연산 낭비를 줄일 수 있고 학습 속도 또한 빨라질 것이라고 예상된다. Overthinking 문제를 해결하는 직관적인 방법은 모델의 깊이를 줄이는 방법이며, 스택 임베딩 인코더 블록의 반복 횟수를 줄여가며 실험들을 진행해보았다. 스택 임베딩 인코더 블록을 3번 반복하는 구조를 '모델 2', 2번 반복하는 구조를 '모델 3', 그리고 1번 반복한 구조를 '모델 4'라고 명명한다. 표 2는 모델 2~4까지의 성능표다.

표 2 각 단계별 벡터를 이용한 캡션의 성능 결과표

표 2-1 모델 2의 성능 결과표

분류	B@1	B@2	B@3	B@4	R	C
O_0	65.7	53.1	44.4	37.2	53.7	37.7
O_1	**75.4**	**63.8**	**54.9**	**47.7**	**62.7**	**43.8**
O_2	73.4	61.5	52.6	45.5	60.9	41.5
O_3	66.8	53.5	43.7	36.3	52.8	28.2

표 2-2 모델 3의 성능 결과표

분류	B@1	B@2	B@3	B@4	R	C
O_0	72.4	59.6	50.3	42.6	58.6	38.2
O_1	**73.5**	**61.5**	**52.5**	**44.8**	**60.5**	**40.5**
O_2	70.6	58.3	49.2	41.7	58.5	37.3

표 2-3 모델 4의 성능 결과표

분류	B@1	B@2	B@3	B@4	R	C
O_0	58.8	46.8	38.6	32.0	50.2	31.0
O_1	**69.1**	**55.9**	**46.9**	**39.6**	**56.8**	**42.4**

스택 임베딩 인코더 블록의 반복 횟수를 줄여가며 학습을 진행했을 때 표 3과 동일하게 모든 모델이 중간의 네트워크의 성능이 좋았고, 중간 네트워크 중 최고 성능을 달성한 이후 성능의 하락이 나타났다. 표 2와 3을 통해서 때에 따라서는 계층 중간의 결과가 최고 성능일 수 있음을 발견했다.

평가 데이터 중 일부를 샘플링하고, 이에 대한 중간 단계 캡션 생성기의 결과를 이용하여 분석을 진행하였다. 표 3은 샘플링한 평가 데이터의 각 단계별 생성된 캡션의 예시이다. 해당 레퍼런스는 형태소 15개 이하인 레퍼런스 중 샘플링한 3개의 레퍼런스를 명시하였다.

표 3 샘플링된 평가데이터에 대한 내부 캡션 생성기의 결과 예시

분류	예시 문장
O_0	길가 에서 차 를 몰 고 있 다 .
O_1	OOV 남자 가 차 를 몰 고 있 다 .
O_2	OOV 사람 이 차 를 몰 고 있 다 .
O_3	차 가 도로 를 달리 고 있 다 .
O_4	차 가 도로 를 따르 아 달리 고 있 다 .
해당 레퍼런스	(1) 빨갛 ㄴ 스포츠카 가 고속도로 를 달리 고 있 다 . (2) 빨갛 ㄴ 차 가 도로 를 달리 고 있 다 . (3) 자동차 가 산길 에서 빠르 게 달리 고 있 다 .

분류	예시 문장
O_0	한 남자 가 무대 에서 노래 를 부르 고 있 다 .
O_1	남자 가 무대 에서 공연 하 고 있 다 .
O_2	남자 가 무대 에서 노래 를 부르 고 있 다 .
O_3	남자 가 노래 를 부르 고 있 다 .
O_4	밴드 가 무대 에서 공연 하 고 있 다 .
해당 레퍼런스	(1) 남자 가 노래 를 부르 고 있 다 . (2) 밴드 가 라이브 관객 을 위 하 아 노래 하 ㄴ다 (3) 가수 가 무대에서 공연 하 고 있 다 .

분류	예시 문장
O_0	레시피를 보이 어 주 는 여 자
O_1	사람 이 음식을 준비 하 고 있 다 .
O_2	사람 이 음식을 준비 하 고 있 다 .
O_3	요리 를 준비 하 는 사람 들
O_4	부엌 에서 음식을 준비 하 는 사람 이 있 다 .
해당 레퍼런스	(1) 요리사 가 요리를 만들 는 중 이 ㅂ니다 (2) 사람 이 요리 를 하 고 있 다 . (3) 누군가 가 그릇 에 재료 를 넣 고 있 다

분류	예시 문장
O_0	브레이커 를 하 는 남자
O_1	한 여성 이 인터뷰 를 하 고 있 다 .
O_2	한 여성 이 인터뷰 를 하 고 있 다 .

O_3	여자 가 말을 하 고 있 다 .
O_4	아시아 여성 이 뉴스 에 대하 아 이야기 하 고 있 다 .
해당 레퍼런스	(1) 두 여자 와 한 남자 가 쇼에서 이야기를 하 고 있 다 . (2) 한 노인 이 두 여자 와 토론을 하 고 있 다 . (3) 남자 와 여자 둘 이 분할 화면에서 이야기를 나누 ㄴ다 .

표 3의 결과를 분석하면, O_4의 결과가 O_1의 결과에 비해 입력된 영상에 대해 구체적인 문장을 생성하는 결과를 보였지만 성능을 측정할 시 오히려 O_4의 결과(구체적인 문장)가 더 낮은 성능을 보였다.

표 3 내 두번째 예시 문장의 O_1와 O_4를 비교하여 설명한다. O_1은 '사람 이 음식 을 준비 하 고 있 다'이고 O_4는 '부엌 에서 음식 을 준비 하는 사람 이 있 다 .'이다. O_0을 제외한 캡션 생성 문장들은 요리(음식)을 준비하는 사람에 대해 캡션을 생성하고 있으나 해당 비디오의 모든 레퍼런스에서 '부엌'이라는 장소가 나타나지 않아 O_4의 성능 하락의 원인이라고 분석된다.

또 다른 예시로, 표 3 내 네번째 예시 문장의 O_1와 O_4를 비교하여 설명한다. O_1는 '한 여성 이 인터뷰 를 하 고 있 다 . '이고 O_4는 '아시아 여성 이 뉴스 에 대하 아 이야기 하 고 있 다 . '이다. 비디오 캡션 생성의 결과로써, 사람이 보기에 O_4의 결과 문장이 더 적절할 수 있다. 하지만 해당 비디오의 모든 레퍼런스에서는 '아시아 여성'이 나타나지 않는다. 또한 모든 레퍼런스 내 존재하지만 이야기하는 대상이 불분명한 '인터뷰(이야기하다)'와 달리 '뉴스'에 대해 이야기한다고 결과를 생성했기 때문에 성능 하락에 영향을 주었을 것이라고 분석된다.

각 층별 결과는 스택 임베딩 인코더 블록을 반복할 수록 입력 비디오에 대해 구체적인 문장을 생성하는 결과를 보였다. 그 이유는 스택 임베딩 인코더 블록이 비디오와 범주 정보에 유용한 자질을 추출하는데, 이 때 반복할수록 더 구체적인 자질을 추출하기 때문이다. 이에 따른 결과로써, 스택 임베딩 인코더 블록을 반복할수록 더 구체적인 문장을 생성하는 결과를 보였다.

성능 측정 지표는 생성된 문장이 레퍼런스와 비교하였을 때 얼마나 유사한 시퀀스를 출력하는지에 대한 것이고, 구체적인 문장이지만 레퍼런스에 포함이 되지 않은 단어들이 포함될 경우 레퍼런스와 유사도가 떨어지기 때문에 이것이 성능 하락의 원인이 된다. 그렇기 때문에 구체적인 문장보다는 일반화한 문장에 대해 성능이 높게 나타나는 것으로 분석된다.

5. 결론

본 논문에서는 각 중간 단계 캡션 생성기를 추가하여 문장을 생성함으로써 모델의 학습이 제대로 되고 있는지 확인하고 모델의 단계별 분석을 진행하였다. 이는 학습 모델의 네트워크가 어느 특정한 단계에서 레퍼런스와 다른 결과를 도출하는지 단계별 결과를 추출하기 위함이다.

주어진 입력 데이터로부터 각 중간 단계 캡션 생성기의 결과는 다르게 나타나고, 최고 성능을 달성한 이후 과한 연산이 오히려 노이즈로 작용하여 오류를 발생시키는 overthinking 문제가 발생함을 확인할 수 있었다. 이를 확인하기 위해 네트워크의 깊이를 줄여가며 실험을 진행했으며, 중간 네트워크 중 최고 성능을 달성한 이후 성능의 하락이 나타났다. 각 중간 단계 캡션 생성기의 결과에 따라 계층 중간의 결과가 최고 성능일 수 있음을 발견했다. 이 결과를 통해 많은 신경망을 쌓는 것보다 적절한 신경망을 쌓는 것이 더 좋다는 것을 확인하였다.

현재 모델에서는 데이터에 대한 형태소 15개 이하라는 제약을 두었다. 이러한 제약 없이 진행할 시에는 현재 제안 구조보다 추가적인 연산이 더 필요할 것으로 예상된다. 이러한 레이어에 대한 설정을 이용하여 입력되는 데이터의 복잡도에 따라 레이어 수를 가변으로 설정하여 학습할 수 있는 모델에 대해 적용이 필요할 것으로 보이며 우리는 이것을 향후 연구로 둔다.

그리고 이런 단계별 캡션에 대한 정보를 학습하는 부분에 반영하여 추가 자질로 사용하는 방법을 연구하고, 중간 단계의 결과를 종합하는 다중 손실함수를 적용한 모델 등 비디오 캡션 모델의 성능을 향상시키기 위한 연구를 진행할 예정이다.

Acknowledgement

본 연구는 한국전자통신연구원 연구운영비지원사업의 일환으로 수행되었음. [19ZH1300, 오픈시나리오 기반 프로그래머블 인터랙티브 미디어 창작 서비스 플랫폼 개발]

참고문헌

[1] Kaya, Y., Hong, S., & Dumitras, T., “Shallow-Deep Networks: Understanding and Mitigating Network Overthinking, In International Conference on Machine Learning, pp. 3301-3310, 2019.

[2] SZEGEDY, Christian, et al.,“Rethinking the inception architecture for computer vision”, In: Proceedings of the IEEE conference on computer vision and pattern recognition, p. 2818-2826, 2016.

[3] 박다솔, 손정우, 김선중, 차정원, “Multi-head Self-Attention을 이용한 비디오 캡션 생성”, 한국정보과학회 학술발표논문집, VOL 46, NO.01, pp.494~0496, 2019.

[4] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T.,“Enriching word vectors with subword information”, Transactions of the Association for Computational Linguistics, Vol 5, pp. 135-146, 2017.

[5] Chollet, F., “Xception: Deep learning with

depthwise separable convolutions, In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1251-1258, 2017.
[6] Kaiser, L., Gomez, A. N., & Chollet, F., “Depthwise separable convolutions for neural machine translation”, arXiv preprint arXiv:1706.03059, 2017.
[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I., “Attention is all you need”, In Advances in neural information processing systems, pp. 5998-6008, 2017.
[8] Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi, H.,“Bidirectional attention flow for machine comprehension”, arXiv preprint arXiv:1611.01603, 2016.
[9] Xiong, C., Zhong, V., & Socher, R.,“Dynamic coattention networks for question answering”, arXiv preprint arXiv:1611.01604, 2016.
[10] Xu, J., Mei, T., Yao, T., & Rui, Y., “Msr-vtt: A large video description dataset for bridging video and language”,In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 5288-5296, 2016.