8. Low-shot Visual Recognition by Shrinking and Hallucinating Features

논문 : https://arxiv.org/pdf/1606.02819.pdf

요약

- 대량의 데이터를 가진 기존 클래스를 학습한 뒤 적은 양의 새로운 클래스가 추가된 few-shot learning을 구현한 논문
- Feature extractor가 대량의 데이터를 학습할 때 generalization할 수 있도록 Squared Gradient Magnitude loss(SGM)을 제안 (일반적인 classification loss와 regularization을 혼용)
- 새로운 클래스에 대한 데이터를 늘리기 위해서 기존데이터를 활용한 generate model를 제안
    - 1,2-shot learning에서는 해당 기법이 정확도를 크게 향상시킴

문제제기

- 일반적으로 image classification문제에서는 대량의 데이터를 가지고 큰 모델을 학습시켜서 정확도를 올리고 있음
- 실제 환경에서는 기존 모델이 새로운 클래스에 대해서 학습해야하는 상황이 있을텐데 데이터 확보가 힘듦
- 저자는 적은 양의 데이터로 기존 클래스 + 새로운 클래스를 분류할 수 있는 기법을 제안
    - 부족한 데이터를 채워줄 수 있도록 기존 데이터를 바탕으로 새로운 클래스의 feature vector를 generate하는 model
    - 기존 클래스를 학습할 때 classification에 사용되는 loss 외에 regularization loss를 추가
- 다른 논문과 다르게 기존 클래스에 대한 정확도 유지를 고려한 실험을 진행

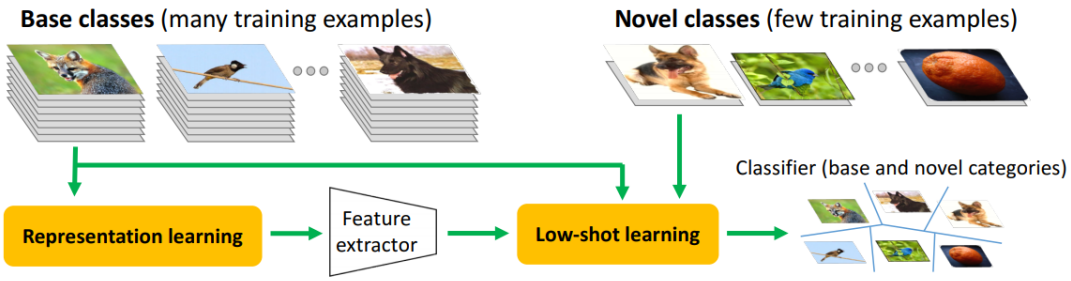

Figure 1: **Our low-shot learning benchmark in two phases: representation learning and low-shot learning.** Modern recognition models use large labeled datasets like ImageNet to build good visual representations and train strong classifiers (*representation learning*). However, these datasets only contain a fixed set of classes. In many realistic scenarios, once deployed, the model might encounter novel classes that it also needs to recognize, but with very few training examples available (*low-shot learning*). We present two ways of significantly improving performance in this scenario: (1) a novel loss function for representation learning that leads to better visual representations that generalize well, and (2) a method for hallucinating additional examples for the data-starved novel classes.

Feature vector generation model (그림 2)

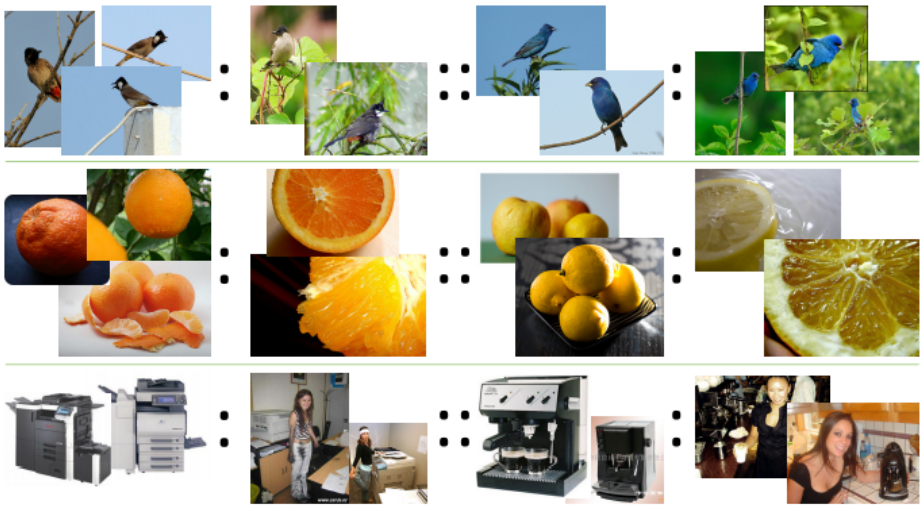

Figure 2: Example mined analogies. Each row shows the four image clusters that form the four elements in the analogy. **Row 1:** birds with a sky backdrop vs birds with greenery in the background. **Row 2:** whole fruits vs cut fruit. **Row 3:** machines (printer, coffee making) in isolation vs the same machine operated by a human.

- 기존 클래스를 학습할 때 사용했던 데이터를 바탕으로 새로운 클래스에 대한 부족한 feature vector를 생성하는 기법을 제안
  - ex) 참새 사진이 있을 때, 참새가 하늘을 나는 사진(배경이 대체로 하늘색)과 나뭇가지에 앉아있는 사진(배경이 대체로 초록/갈색)이 같은 클래스라는 성격을 이용해서 비둘기가 하늘을 나는 사진으로 나뭇가지에 앉아있는 사진의 feature vector를 generate하는 것
- 데이터 구조
  - 기존 클래스의 feature representation을 100개의 cluster로 분리
  - 같은 클래스인 cluster centroid 두개를 페어로 묶음
    - ex) a 클래스로 분류된 클러스터 두개 c_1^a, c_2^a
  - 묶음 두개를 하나의 데이터로 사용
    - $(c_1^a, c_2^a, c_1^b, c_2^b)$
  - 모델의 입력
    - concat한 데이터 (c_1^a, c_1^b, c_2^b)
  - 모델의 출력
    - feature representation $\hat{c}_2^a = G([c_1^a, c_1^b, c_2^b])$
  - 모델의 objective function
    - $\lambda L_{mse}(\hat{c_2^a}, c_2^a) + L_{cls}(W, \hat{c_2^a}, a)$ 를 최소화
    - $L_{mse}(\hat{c_2^a}, c_2^a)$: 모델이 생성한 feature vector와 실제 feature vector에 대한 mean squared error
    - $L_{cls}(W, \hat{c_2^a}, a)$: 모델이 생성한 feature vector로 classifier W가 정답 클래스에 대한 확률
- 모델 자체는 Fully Connected Layer 3개를 붙인 것으로 실험

Squared gradient magnitude loss (SGM)

기호	설명

ϕ	Feature Extractor
W	Classifier
D	Large Dataset Feature extractor 학습에 사용 ex) ImageNet
S	Small Dataset few-shot learning에 사용
δ_{yk}	k번째 클래스가 정답일 때 1, 아닌경우 0

$$\min_{W,\phi} L_D(\phi, W) = \min_{W,\phi} \frac{1}{|D|} \sum_{(x,y)\in D} L_{cls}(W, \phi(x), y) \quad (1)$$

$$L_{cls}(W, x, y) = -\log p_y(W, x) \quad (2)$$

$$p_k(W, x) = \frac{\exp(w_k^T x)}{\sum_j \exp(w_j^T x)}. \quad (3)$$

- Feature Extractor + Classifier 학습
  - ImageNet과 같은 Large Dataset D 으로 Feature Extractor를 학습
    - Feature Extractor는 ResNet과 같은 모델 구조를 사용
    - Classifier는 ResNet 뒤에 붙는 Fully Connected Layer
  - 식 1~3 : 기존의 Supervised learning에서 사용하는 Objective function 설명
    - Large Dataset D에 대한 multiclass logistic loss
    - 식 3 : feature representation x가 입력됐을 때 Fully connected layer 값으로 softmax

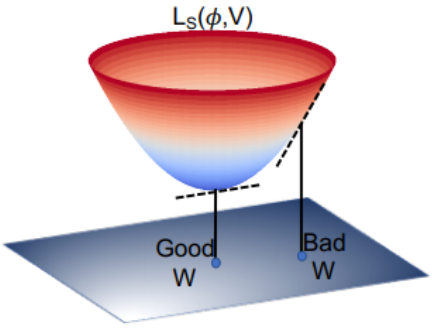

Figure 3: Motivation for the SGM loss. We want to learn a representation ϕ such that the arg min of the small set training objective $L_S(\phi, V)$ matches W, the classifier trained on a large dataset D.

$$\min_V L_S(\phi, V) = \min_V \frac{1}{|S|} \sum_{(x,y)\in S} L_{cls}(V, \phi(x), y) \quad (4)$$

$$\nabla_V L_S(\phi, V) = [g_1(S,V), \dots g_K(S,V)] \quad (6)$$

$$g_k(S,V) = \frac{1}{|S|} \sum_{(x,y)\in S} (p_k(V, \phi(x)) - \delta_{yk})\phi(x) \quad (7)$$

- 기존의 학습 과정 (식 1~3)에서 Small Dataset S을 고려하여 기본 클래스에대한 low-show learning을 시뮬레이션
  - $S \subset D, |S| \ll |D|$: Large Dataset D에서 random sampling으로 만든 small dataset S
  - 식 4 : Small Dataset S에 대한 Classifier V의 objective function
  - 저자들은 W와 V를 동일시 할수 있도록 Classifier W와 Classifier V가 같은 objective function을 가지도록함
    - 바꿔말하면 W가 minimize 될수록 V가 minimize되기를 바람
    - V와 W가 같은지 확인하려면?
      - 그림 3과 같이 V의 Training loss $L_S(\phi, V)$ 가 Convex하다고 가정
      - $L_S(\phi, V)$ 의 기울기 $\nabla_V L_S(\phi, V)$ 가 0이면 같은 성능이라고 할 수 있음
      - 식 6~7 : Small Dataset S에 K개의 클래스가 존재할 때, 각 클래스별 기울기 모음 (식 6, 7 풀이 : http://home.bharathh.info/lowshotsupp.pdf)
    - W로 analytical function을 표현해보면?
      - $\tilde{L}_S(\phi, W) : \frac{1}{|S|^2} \sum_{k=1}^{K} \| \sum_{(x,y)\in S} (p_k(W, \phi(x)) - \delta_{yk})\phi(x) \|^2$
      - 위 함수를 single example (x,y)에 대해 적용해보면 식 8~9로 표현 가능
        - 식 9 : 잘못 분류된 데이터 포인트에 대해 더 높은 가중치를 주는 식을 표현
        - Loss는 feature activation에 대한 가중치가 적용된 L2 Regularization을 뜻함

$$\tilde{L}_S(\phi, W) = \sum_{k=1}^{K} (p_k(W, \phi(x)) - \delta_{yk})^2 \|\phi(x)\|^2 \quad (8)$$

$$= \alpha(W, \phi(x), y)\|\phi(x)\|^2. \quad (9)$$

- Square Gradient Magnitude (SGM)은 Large Dataset D의 모든 데이터에서 평균을 내는 최종 식으로 정리 (식 10)
  - feature regularization이 학습에 도움이 되는 이유는?
    - 기존의 supervised learning에서 사용하는 loss는 feature extractor가 feature representation을 잘 못만들더라도 이를 적절히 무시할 수 있도록 classifier 기준의 Loss를 사용
    - SGM을 추가해서 feature extractor가 더 다양한 표현을 배우기 위함
    - 작은 데이터셋은 클래스별로 충분한 학습을 할 수 없는 데이터이므로 잘 표현하는 feature extractor가 있으면 classifier가 더 빨리 수렴할 수 있음을 논문이 주장
  - minibatch를 사용하는 경우엔 어떻게 처리?
    - λ로 decay를 더 줘서 batch별 classification loss와 합해서 사용
  - SGM 대신 사용할 수 있는 feature regularization
    - $\alpha(W, \phi(x), y) \in [0, 2]$ 이므로 SGM이 너무 커져서 classification loss를 무시할 수 있음
    - unsupervised learning에서 사용하는 방법처럼 간단하게 L2 norm이나 L1 norm으로 대체할 수 있음
    - SGM으로 어느정도 학습하다가 triplet loss로 단계적 학습을 하는 방법도 있음 (실험결과 기준으로 제안 기법보단 성능이 떨어짐)
- 논문은 Supervised Learning에서 사용하는 classification logistic loss에 SGM을 더한 Loss를 학습에 사용 (식 11)
  - 식 11을 바탕으로 Feature Extractor를 학습 한 뒤, few-shot learning에서는 파라미터를 프리징함 (transfer learning과 유사)

$$L_D^{SGM}(\phi, W) = \frac{1}{|D|} \sum_{(x,y)\in D} \alpha(W, \phi(x), y)\|\phi(x)\|^2 \tag{10}$$

$$\min_{W,\phi} L_D(\phi, W) + \lambda L_D^{SGM}(\phi, W) \tag{11}$$

$$p_k(W, x) = \frac{\exp(w_k^T x)}{\sum_j \exp(w_j^T x)}.$$

$$L_{cls}(W, x, y) = -\log p_y(W, x) \qquad \alpha(W, \phi(x), y) = \sum_k (p_k(W, \phi(x)) - \delta_{yk})^2$$

$$\min_{W,\phi} \frac{1}{|D|} \sum_{(x,y)\in D} L_{cls}(W, \phi(x), y) + \lambda \frac{1}{|D|} \sum_{(x,y)\in D} \alpha(W, \phi(x), y)\|\phi(x)\|^2$$

$$\min_{W,\phi} L_D(\phi, W) + \lambda L_D^{SGM}(\phi, W)$$

실험

- ImageNet 1K 데이터셋으로 성능 비교
  - feature extractor의 구조는 ResNet을 사용

Representation	Lowshot phase	n=1	2	5	10	20
ResNet-10						
Baseline	Classifier	14.1	33.3	56.2	66.2	71.5
Baseline	Generation* + Classifier	29.7	42.2	56.1	64.5	70.0
SGM*	Classifier	23.1	42.4	*61.7*	69.6	*73.8*
SGM*	Generation* + Classifier	*32.8*	46.4	*61.7*	*69.7*	*73.8*
Batch SGM*	Classifier	23.0	42.4	61.9	**69.9**	**74.5**
L1*	Classifier	20.8	40.8	59.8	67.5	71.6
L2*	Classifier	29.1	*47.4*	**62.3**	68.0	70.6
Triplets	Classifier	24.5	41.8	56.0	61.3	64.2
Dropout [20]	Classifier	26.8	43.9	59.6	66.2	69.5
Decov [7]	Classifier	13.0	33.9	59.3	68.9	73.4
Multiverse [30]	Classifier	13.7	30.6	52.5	63.8	71.1
Baseline	Data augmentation	16.0	31.4	52.7	64.4	71.8
Baseline	Model Regression [47]	20.7	39.4	59.6	68.5	73.5
Baseline	Matching Network [46]	**41.3**	**51.3**	*62.1*	67.8	71.8
Baseline-ft	Classifier	12.5	29.5	53.1	64.6	70.4
ResNet-50						
Baseline	Classifier	28.2	51.0	71.0	*78.4*	*82.3*
Baseline	Generation* + Classifier	*44.8*	**59.0**	*71.4*	77.7	*82.3*
SGM*	Classifier	37.8	57.1	**72.8**	**79.1**	**82.6**
SGM*	Generation* + Classifier	**45.1**	*58.8*	*72.7*	**79.1**	**82.6**

Table 1: Top-5 accuracy on only novel classes. Best are bolded and blue; the second best are italicized and red. *Our methods.

Representation	Lowshot phase	n=1	2	5	10	20
ResNet-10						
Baseline	Classifier	43.0	54.3	67.2	72.8	75.9
Baseline	Generation* + Classifier	52.4	59.4	67.5	72.6	76.9
SGM*	Classifier	49.4	60.5	*71.3*	**75.8**	*78.1*
SGM*	Generation* + Classifier	*54.3*	*62.1*	*71.3*	**75.8**	*78.1*
Batch SGM*	Classifier	49.3	60.5	71.4	*75.8*	**78.5**
L1*	Classifier	47.1	58.5	69.2	73.7	76.1
L2*	Classifier	52.7	**63.0**	**71.5**	74.8	76.4
Triplets	Classifier	47.6	57.1	65.2	68.4	70.2
Dropout [20]	Classifier	50.1	59.7	68.8	72.7	74.7
Decov [7]	Classifier	43.3	55.7	70.1	75.4	77.9
Multiverse [30]	Classifier	44.1	54.2	67.0	73.2	76.9
Baseline	Data Augmentation	44.9	54.0	66.4	73.0	77.2
Baseline	Model Regression [47]	46.4	56.7	66.8	70.4	72.0
Baseline	Matching Network [46]	**55.0**	61.5	69.3	73.4	76.2
Baseline-ft	Classifier	41.7	51.7	65.0	71.2	74.5
ResNet-50						
Baseline	Classifier	54.1	67.7	*79.1*	*83.2*	**85.4**
Baseline	Generation* + Classifier	*63.1*	**71.5**	78.8	82.6	**85.4**
SGM*	Classifier	60.0	*71.3*	**80.0**	**83.3**	*85.2*
SGM*	Generation* + Classifier	**63.6**	**71.5**	**80.0**	**83.3**	*85.2*

Table 2: Top-5 accuracy on base and novel classes. Best are bolded and blue; the second best are italicized and red. *Our methods.

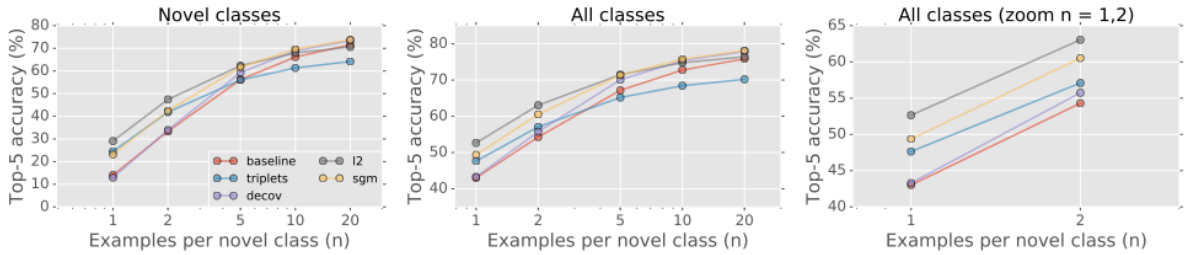

Figure 4: **Representation learning comparison.** Top-5 accuracy on ImageNet1k val. Top-performing feature regularization methods reduce the training samples needed to match the baseline accuracy by 2x. Note the different Y-axis scales.

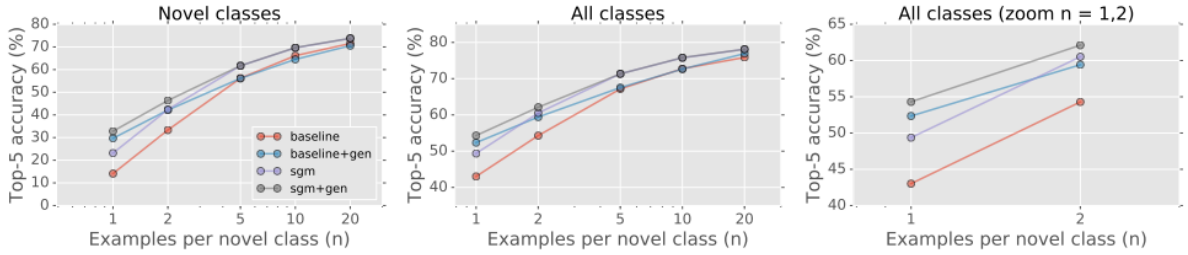

Figure 5: **Comparisons with and without example generation.** Top-5 accuracy on ImageNet1k val. Note the different Y-axis scales.

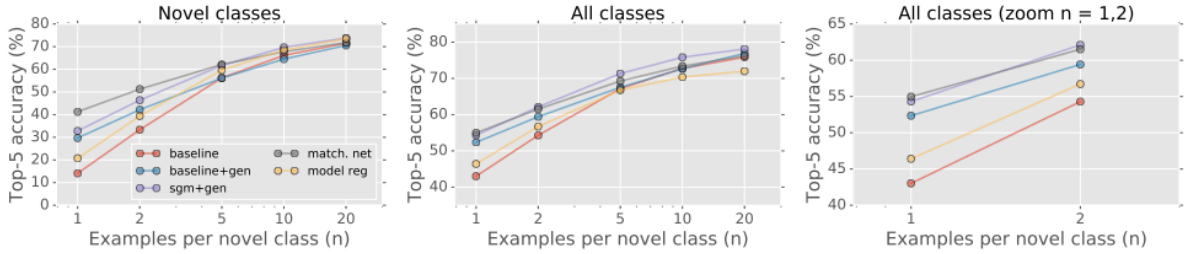

Figure 6: **Comparison to recently proposed methods.** Top-5 accuracy on ImageNet1k val. Note the different Y-axis scales.