변분 오토인코더(variational autoencoder)와 k-means 알고리즘을 활용한 웨이퍼 결함패턴 클러스터링 알고리즘 개발

2019.04~2019.05

github: https://github.com/joyoon1110/wafer_clustering

연구 노트

연구 목표

반도체 제조공정은 공정편차 또는 기타생산문제로 인하여 제품에 문제가 발생하기 쉽다. 모든 공정 단계의 품질을 보증하기 위해 웨이퍼 테스트 값 측정은 제조공정에서 근본원인을 찾는데 중요하다. 자동화된 검사와 여러 공정 단계에서 얻어진 웨이퍼맵(wafermap) 데이터 패턴 인식은 초기 생산 문제를 발견하여 제조의 효율성을 향상시키는데 중요한 잠재력을 가지고 있다. 웨이퍼맵 측정 데이터에 패턴의 비지도 학습 방법(unsupervised learning)인 클러스터링(clustering)을 사용하는 머신러닝 접근 방법을 제안하고, 실제 데이터 셋에 제안한 방법의 성능을 실험적으로 평가한다.

개념

연구 노트

연구 목표

K-means Clustering 평가방법

개념

Review on determining number of Cluster in K-means Clustering

<**Elbow method**>

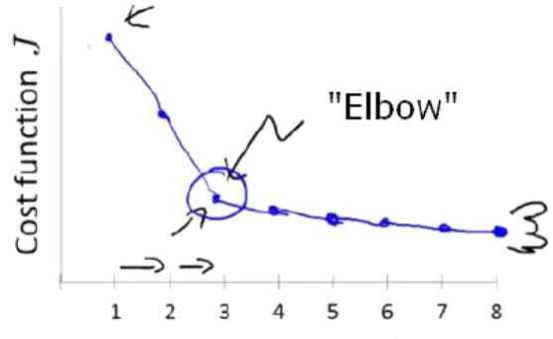


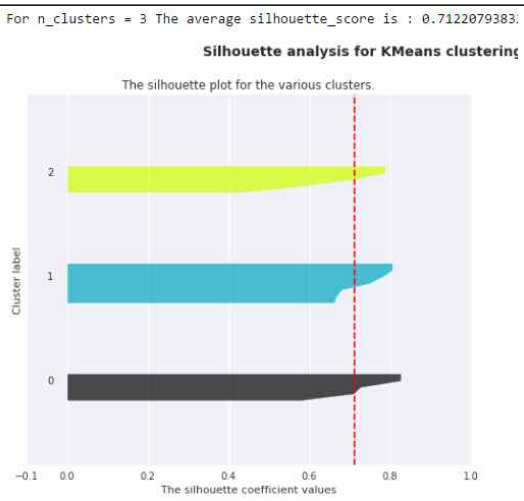
Fig. 1 identification of Elbow point

K-means clustering의 성능을 비교하기 위해 클래스 내 SSE(왜곡)을 사용함.

클래스 내 SSE를 바탕으로 엘보우 방법이라고 하는 그래프를 사용하여 문제에 최적인 클러스터 개수 k를 추정한다.

k 값을 바꾸어 가며 왜곡이 빠르게 증가하는 지점의 k 값을 찾는 것이다.

<**Silhouette score**>



실루엣 분석은 클러스터 내 샘플들이 얼마나 조밀하게 모여 있는지를 측정하는 그래프 도구이다.

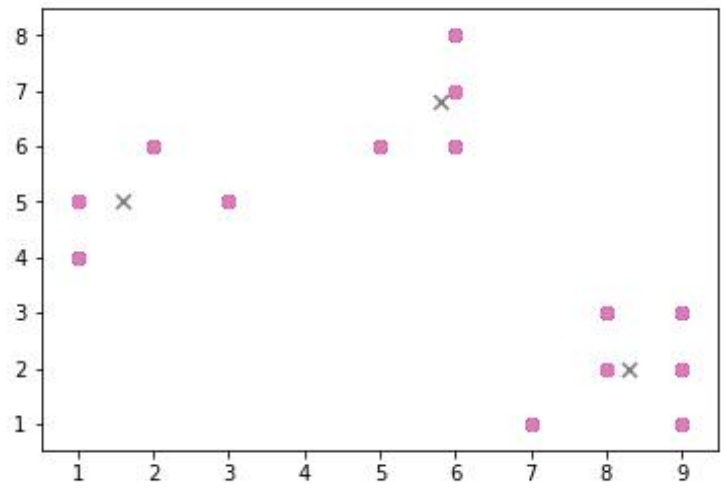
실루엣 계수는 -1과 1 사이 값을 가진다. 클러스터 응집력과 분리도가 같으면 실루엣 계수가 0이 된다. 이상적인 실루엣 계수는 1에 가깝게 된다.

연구 노트

연구 목표

clustering 결과를 향상시키기 위한 방법

개념



초기 센트로이드가 좋지 않게 선택되면 나쁜 군집 결과를 만들거나 수렴이 느려진다. 이 문제를 해결하는 방법은 같은 데이터 셋에서 k-means 알고리즘을 여러 번 실행하여 SSE입장에서 가장 성능이 좋은 모델을 선택하는 것이다.

또 다른 방법은 K-means++ 알고리즘을 통해 초기 센트로이드가 서로 멀리 떨어지도록 위치시키는 것이다. 이는 기본 K-mean보다 일관되고 훌륭한 결과를 만든다.

1. 선택한 k개의 센트로이드를 저장할 빈 집합 M을 초기화한다.
2. 입력 샘플에서 첫 번째 센트로이드 $\mu^{(i)}$를 랜덤하게 선택하고 M에 할당한다.
3. M에 있지 않은 각 샘플 $\mu^{(i)}$에 대해 M에 있는 센트로이드까지 최소 제곱 거리 $d(x^{(i)}, M)^2$을 찾는다.
4. 가중치가 적용된 확률 분포를 사용하여 다음 센트로이드 $\mu^{(p)}$를 랜덤하게 선택한다.
5. K개의 센트로이드를 선택할 때까지 단계 2와 3을 반복한다.
6. 그다음 기본 K-means 알고리즘을 수행한다.

연구 노트

연구 목표

K-means와 다른 방식의 clustering 방법

개념

앙상블 방법 또는 다른 군집 알고리즘 적용

1. k-means <-> Hierarchical clustering

2. Unsupervised Wafermap Patterns Clustering via Variational Autoencoders(IJCNN, 2018)

-Variable Autoencoders

<**hierarchical clustering**>

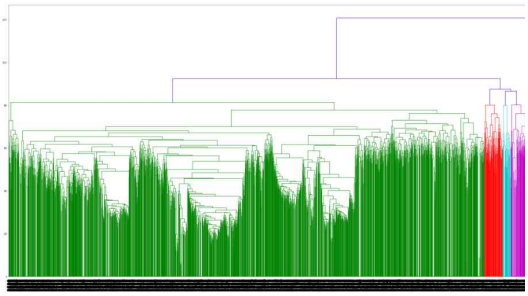
병합 계층 군집

각 샘플이 독립적인 클러스터가 되고 하나의 클러스터가 남을 때까지 가장 가까운 클러스터를 합친다.

분할 계층 군집

분할 군집에서는 전체 샘플을 포함하는 하나의 클러스터에서 시작하여 더 작은 클러스터로 반복적으로 나눈다. 클러스터 안에 샘플이 하나만 남을 때까지 계속한다.

<**덴드로그램**>



장점

1. 덴드로그램(이진 트리 형태로 계층 군집을 시작화할 수 있는 도구)를 그릴 수 있음.

2. 클러스터 개수를 미리 지정할 필요가 없음.

연구 노트

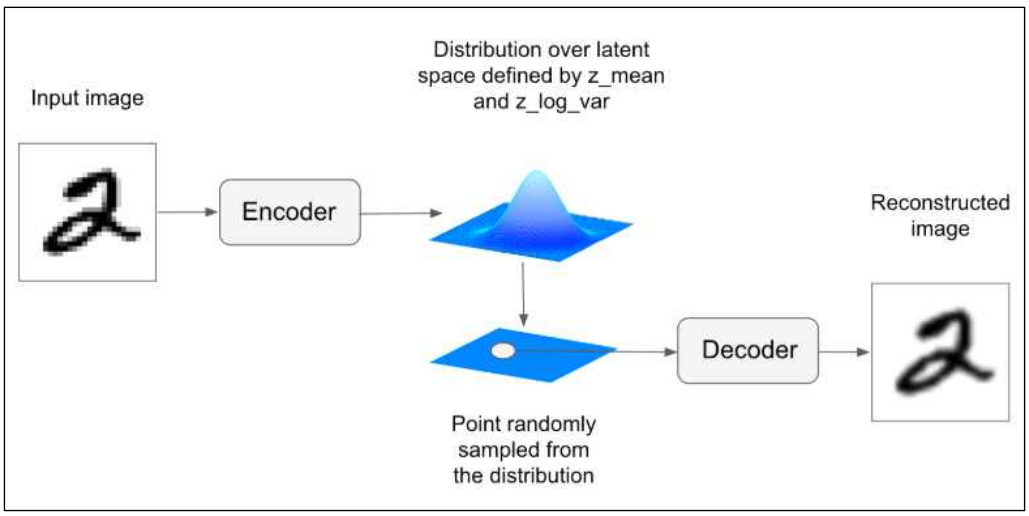
연구 목표

차원축소를 위한 오토인코더 조사

개념

1. Unsupervised Wafermap Patterns Clustering via Variational Autoencoders(IJCNN, 2018)

-Variable Autoencoders



변이형 오토인코더는 생성 모델의 한 종류로 개념 벡터를 사용하여 이미지를 변형하는 데 아주 적절하다. 오토인코더는 입력을 저차원 잠재 공간으로 인코딩한 후 디코딩하여 복원하는 네트워크이다.

고전적인 오토인코더는 이미지를 입력받아 인코더 모듈을 사용하여 잠재 벡터 공간으로 맵핑한다. 그 다음 디코더 모듈을 사용해서 원본 이미지와 동일한 차원으로 복원하여 출력한다. 오토인코더는 입력 이미지와 동일한 이미지를 타깃 데이터로 사용하여 훈련한다. 원본 입력을 재구성하는 방법을 학습하는 것이다.

*Diederik P. Kingma and Max Welling, "Auto-Encoding Variational Bayes, arXiv (2013), https://arxiv.org/abs/1312.6114

연구 노트

연구 목표

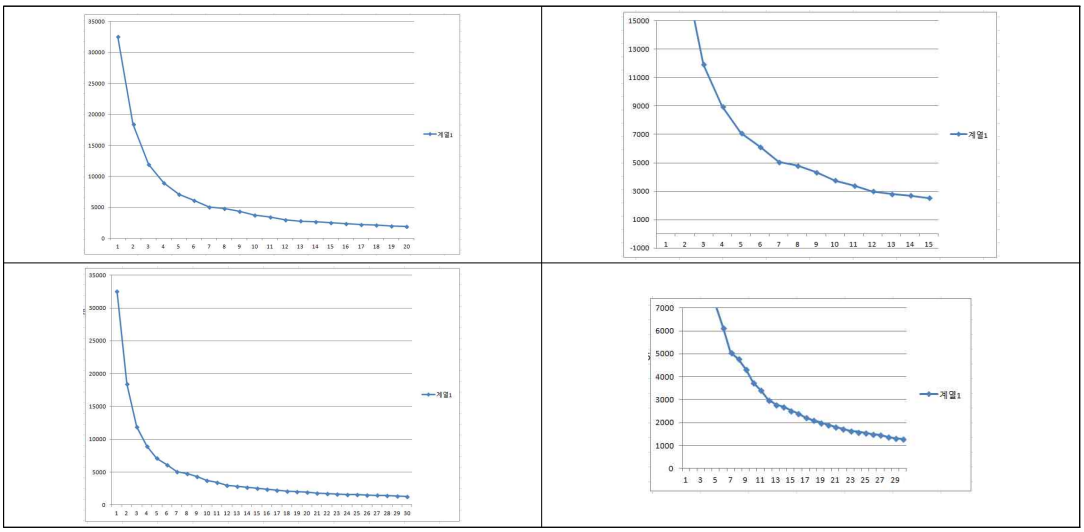
Auto k 값을 찾기 위한 실험 및 평가

개념

Resource 절약 관련 실험

1. 데이터 셋 숫자에 따른 k값이 다름 -> 당연한 문제!
2. auto k 값이 아님 -> angle cosine

```
(array([32527.49023438, 18408.74804688, 11912.46875   ,  8960.16601562,
        7102.95214844,  6124.2109375 ,  5062.31787109,  4796.90039062,
        4336.49560547,  3753.84570312,  3410.53979492,  2999.51049805,
        2795.54858398,  2693.56347656,  2533.00927734,  2390.21826172,
        2215.33154297,  2111.49536133,  2006.41894531,  1913.52307129,
        1813.92919922,  1732.51342773,  1647.51513672,  1596.60913086,
        1554.22412109,  1490.66870117,  1455.63647461,  1384.30688477,
        1315.85168457,  1279.10961914,  1230.61499023,  1196.85632324,
        1173.30639648,  1156.171875  ,  1110.02709961,  1075.21203613,
        1046.21972656,  1025.2520752 ,  1001.12023926,   982.30834961]), array([33, 37, 32, 26, 29, 39, 31, 23, 35, 36,
17,  7, 12, 21, 19,
       10, 15, 28,  5,  4,  3,  2,  6, 11,  8,  9, 40, 20, 14, 16, 18, 22,
       25, 27, 30, 34, 38,  1]))
```



연구 노트

연구 목표

variational autoencoder 네트워크구조변경, 파라미터최적화

개념

variational autoencoder 네트워크 구조 변경 및 최적 파라미터

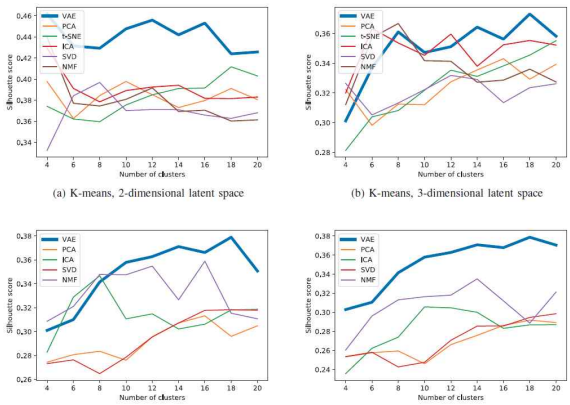
1. 구조 변경

20, 30, 40 K-means의 k값을 다르게 하여도 angle cosine이 일정한 값이 나오지 않음 (범위 내에서 최적값을 찾기 때문으로 판단)

ex) 20->16~18, 30->22~29, 40->35~38

2. Variational Autoencoders -> Wafermap(6종)

Clustering 결과, NMF, PCA, ICA, SVD 중에서 NMF가 그나마 silhouette score값이 높고, VAE가 가장 높은 silhouette score를 가짐.



(a) K-means, 2-dimensional latent space

(b) K-means, 3-dimensional latent space

(c) K-means, 4-dimensional latent space

(d) K-means, 5-dimensional latent space

연구 노트

연구 목표

클러스터링 결과값 도출 및 학습 속도 개선

개념

추가사항 진행(속도 확인 및 클러스터링 관련 VaDE 알고리즘)

1. 클러스터링 결과 같은값 나오는 것에 대한 사항
2. 속도 개선

학습속도(test_dataset) GPU 100회 1분 30초

	A	B	C	D	E	F
1		17_1	17_2	차	편차	편차^2
2	0	390	397	7	14.88235	221.4844
3	1	487	488	1	20.88235	436.0727
4	2	470	465	5	16.88235	285.0138
5	3	1013	1071	58	-36.1176	1304.484
6	4	560	586	26	-4.11765	16.95502
7	5	257	260	3	18.88235	356.5433
8	6	783	795	12	9.882353	97.6609
9	7	592	507	85	-63.1176	3983.837
10	8	1241	1263	22	-0.11765	0.013841
11	9	344	361	17	4.882353	23.83737
12	10	103	104	1	20.88235	436.0727
13	11	575	560	15	6.882353	47.36678
14	12	622	607	15	6.882353	47.36678
15	13	735	717	18	3.882353	15.07266
16	14	959	992	33	-11.1176	123.6021
17	15	407	359	48	-26.1176	682.1315
18	16	462	468	6	15.88235	252.2491
19			합계	372		489.9862
20			평균	21.88235		
21			표준편차	22.1		

K-means 클러스터링 알고리즘에서 K값으로 17을 주고 같은 결과 값이 나오는 지 반복실험을 하였음. 1차와 2차 평가에서 평균 21개의 오차가 발생함.

-> SEED 값을 고정하여 해결하는 방법 구상 및 코드 수정

GPU(P100)으로 100epoch training 하는데 약 90s의 시간이 소요됨.

연구 노트

연구 목표

정확도 개선 작업 및 속도 테스트

개념

1. VaDE 환경구축

-Anaconda에서 실행(O)->시간..

2. Variational Autoencoder

- Wafer clustering

-정확도 개선 추가 작업 및 속도 테스트

*****Variational Autoencoder*****

1) wafer 이미지 적용

cluster number

이미지 전 처리 방법 // 20, 30, 40, 50 // Batch Normalization 유무

30epoch 학습모델 사용

30 Epoch 학습모델(배치X)	30 Epoch 학습모델(배치O)
20->13 30->27 40->39	20->19 30->19 40->39 50->47

	CPU(192.168.0.173)		GPU(192.168.0.173)
	Tensorflow	scikit-learn	tensorflow
Data load (8000장 기준)	8s	←	1s
Training	9s / epoch	←	1.5s / epoch
Elbow method (cosine angle)	30.9s	←	30.4s
k-means clustering	647ms	14.2s	581ms

K-means 알고리즘에서 최대값을 주고 Auto K값을 찾기 위한 실험을 진행함. 배치 유무에따라 다른 결과값(K)이 나오는 것을 확인함.

알고리즘의 순서는 '데이터 로드', 'Training', 'Elbow method', 'cosine angle', 'k-means' 은 순서로 진행하여 위의 표와 같은 시간이 소요되는 것을 확인하였음.

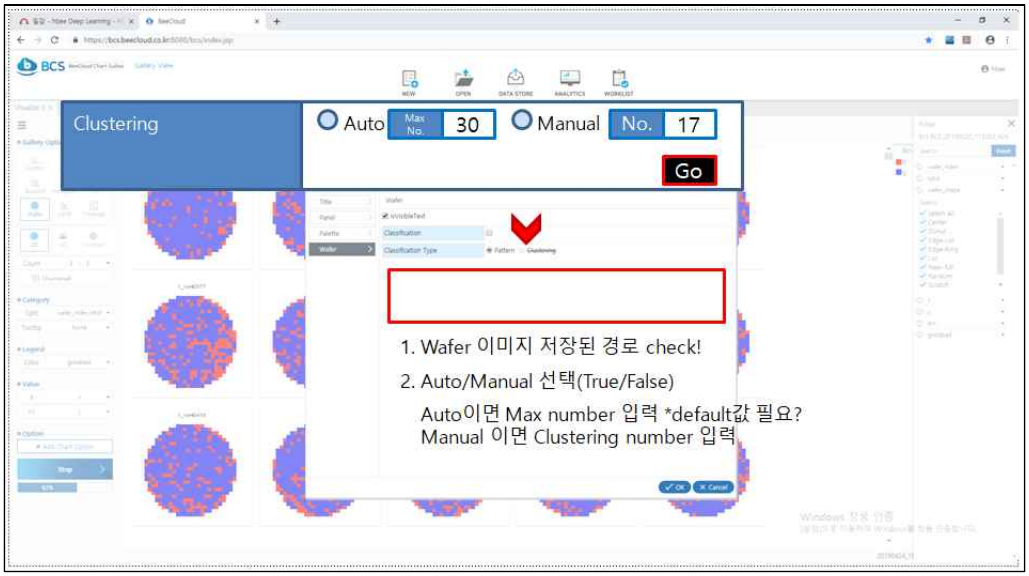
연구 노트

연구 목표

Clustering 기능 추가

개념

1. Clustering 기능 추가



제품에 클러스터링 기능을 구현하기 위해 스크립트를 바탕으로 UI를 구성하였음. Auto/Manual에 따라 K-means의 동작을 다르게 구현함.

사용자는 Auto로 K값을 찾거나 Manual 로 K값을 지정하여 클러스터링의 수를 선택할 수 있도록 함.

연구 노트

연구 목표

실제로 프로그램 배포를 위한 서버에서 속도 테스트를 수행한다.

개념

1. Clutering 기능 추가

```
!python scipt_test.py --image_path='/home/hbee.dong/chip_data/new_version_2019_04_01/black_white/test/' \
--jobid=1111 --auto=True --max_num=20 --cluster_number=30
```

```
이미지 리스트 길이, 생산날짜 비교 (wafer_0000_190430(생산날짜).npy)
1) 이미지 리스트 길이 검사
2) 생산날짜 비교
if not in:
 start training
 save "wafer_0000_190430(생산날짜).npy)
else:
 기존에 저장된 "wafer_0000_190430(생산날짜).npy" 값 사용
```

```
                                        1. Auto
인코더 학습 → 학습 모델(z_mean, z_log_var) → K-means(Elbow method & Cosine transformation) → Auto K → K-means Clustering
              "wafer_8207_190430(생산날짜).npy" 저장          최대 K 값 입력(*default max_num)          1) k_value 2) class_idx 3) distance
                                        2. Manual
                                                  K 값 입력                          → K-means Clustering
                                                                                       1) k_value 2) class_idx 3) distance
```

173 서버 (CPU)

	Load image	Training	K-means
8,000장	0.7 GB (4s)	0.6~0.9 GB 9s / epoch	0.01GB auto: 37~40s manual: 0.6~0.7s
32,000장	4.0 GB (53s)	1.1~1.2 GB 35.6s / epoch	

1) 서버에서 작업 및 테스트

실제로 구동하기 위한 스톰서버에서 data load, training, k-mean 알고리즘을 실행.

8000장 기준

데이터 로드 0.7 GB(4s) / 트레이닝 0.6~0.9 GB(9s/perepoch) / k-mean 37~40s 소요

32000장기준

데이터 로드 4.0 GB(53s) / 트레이닝 1.1~1.2 GB(35s/perepoch) / k-mean 동일 소요

2) 서버 관련 작업 진행

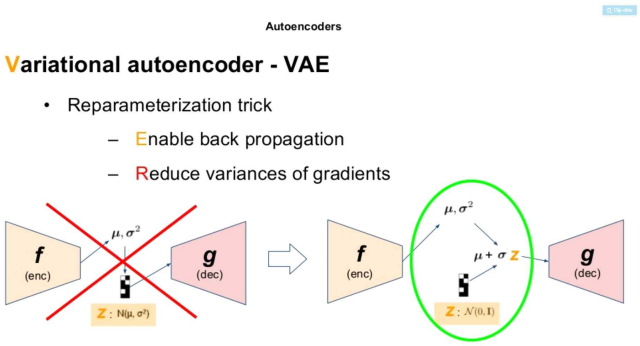
연구 노트

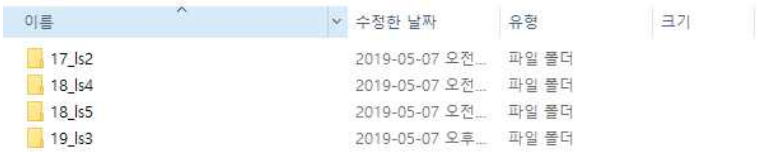
연구 목표

Latent space에 따른 VAE clustering 결과 값 비교를 위한 추가 실험을 진행한다.

개념

1. latent space





Auto K의 최대값이 커지면 최적 K값이 커지는 문제

latent space = 5

20 -> 18

30 -> 23

40 -> 26

50 -> 31

*P100 에서 Elbowmethod & angle cosine 시간외에 동일

2. cluster labeling 연구시작 및 자료 조사

연구 노트

연구 목표

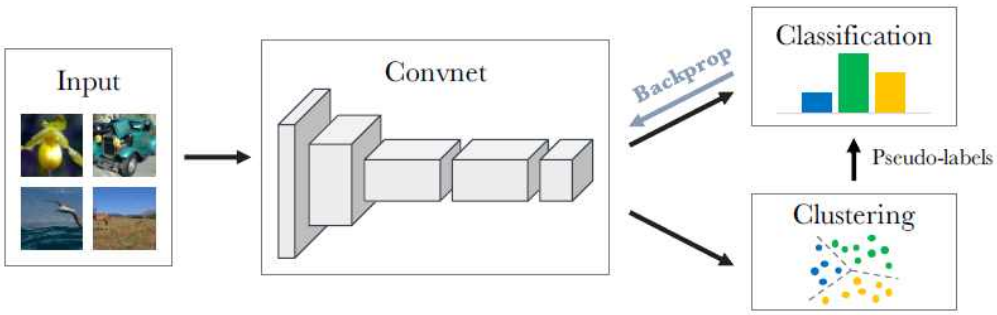
클러스터링 결과 값을 바탕으로 분류기의 정확도를 높이기 위한 방법에 대한 자료 수집 및 실험을 진행한다.

개념

1. Deep Clustering

$$\min_{\theta,W} \frac{1}{N} \sum_{n=1}^{N} \ell\left(g_W\left(f_\theta(x_n)\right), y_n\right)$$

$$\min_{C \in \mathbb{R}^{d \times k}} \frac{1}{N} \sum_{n=1}^{N} \min_{y_n \in \{0,1\}^k} \|f_\theta(x_n) - C y_n\|_2^2 \quad \text{such that} \quad y_n^\top 1_k = 1$$



DeepCluster 는 식 (2)를 이용해 가짜-라벨을 구하고 식 (1)에 의해 가짜-라벨을 잘 예측하는 구조.

학습된 CNN의 특징으로 다시 클러스터링한 후 가짜 라벨링을 변경하고, 변경된 가짜-라벨을 잘 예측하도록 CNN을 업데이트하는 것.

두 과정을 수렴할 때까지 반복한다.

[참고문헌]

Deep Clustering for Unsupervised Learning of Visual Features
Mathilde Caron et al. Facebook AI Research 2018