Anomaly Detection

1. Anomaly Detection and Segmentation이란?

Anomaly란 정상의 범주에 벗어나 있는 모든 것들을 말한다. 따라서 Anomaly Detection이란 불량 검출, 이상 감지 등 비 정상 여부를 탐지하는 기술을 말한다.
Test Instance에 대해 Normal 혹은 Anomaly로 구분해야 하기에 one class classification이라고도 불린다. Anomaly Segmentation은 영상이 입력으로 왔을 때, pixel level로 비정상 여부를 탐지하는 기술을 말한다. 즉 영상의 어떤 부분에 이상치가 위치하고 있는지 localize하는 것을 말한다.

2. Anomaly Detection의 Dataset

1. Classification Dataset

만약 특정 클래스만 정상으로 취급하고 나머지 클래스는 모두 이상치로 취급하면 ImageNet이 대표적인 Anomaly Dataset이다. 예를 들어 '고양이'를 정상 데이터로 학습하여 '개', '비행기' 등을 anomaly로 검출하는 문제에 해당한다.

ex) MNIST, Fashion-MNIST, CIFAR10, ImageNet 등

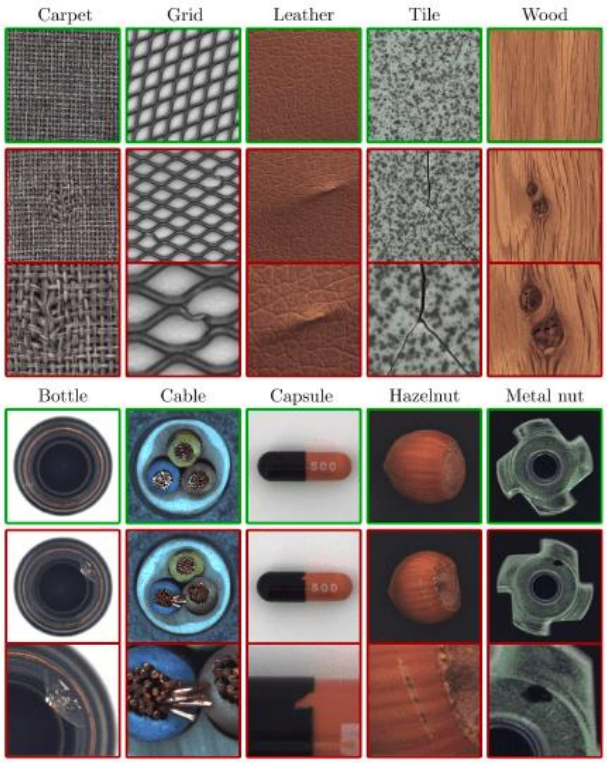
2. Synthetic Dataset

일반 영상이 아닌 깨지거나 비오는 영상 등을 이상치로 취급하면 ImageNet-C가 대표적인 Anomaly Dataset이다. 이는 Image classification을 위한 이미지 데이터 셋에 corrupion을 추가한 데이터셋이다.
ex) ImageNet-C, ImageNet-P, Mnist-C 등

3. Real World Dataset

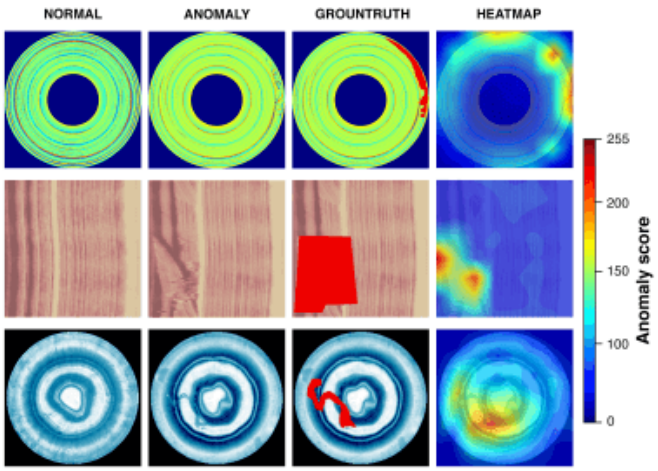
실제 환경에서 발생하는 이상치를 다루는 Dataset으로 mvTec, BTAD, LAG, STC가 대표적인 Anomaly Dataset이다.

3-1. mvTec Dataset



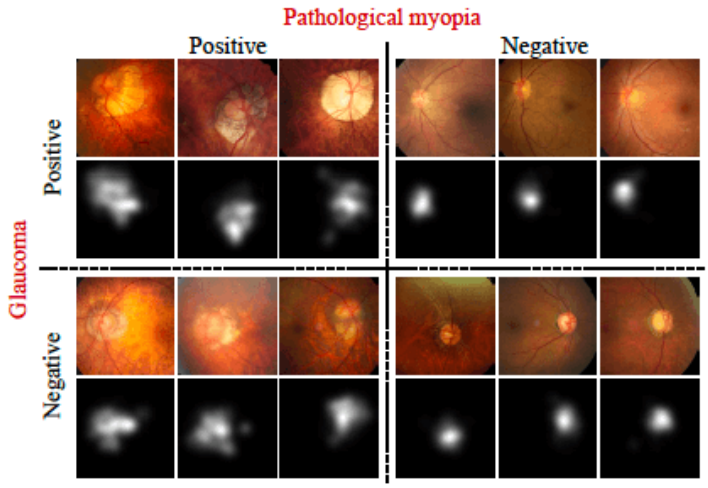
제조 환경에서의 불량 검출을 위한 데이터 셋으로 총 15가지의 카테고리로 분류되어 있다.
3629장의 train 데이터와 1725장의 test데이터로 구성되어 있다.
train 데이터는 normal 데이터만으로, test 데이터는 normal, abnomal 데이터로 구성되어 있다.
15가지의 카테고리 중 bottle, cable 등의 10가지는 object의 특징을 갖고 있고, carpet, grid 등의 5가지는 texture의 특징을 갖고 있다. segmentation을 위한 label도 갖고 있어 detection, segmentation 성능 측정이 가능하다.

3-2. BTAD(Bean Tech Anmaly Detection) Dataset



mvTec과 마찬가지로 제조 환경에서의 불량 검출을 위한 데이터셋이다. product 1,2,3로 구분되어 있으며 각각 400장, 1000장, 399장의 학습 데이터를 포함하고 있다. segmentation을 위한 label도 갖고 있어 detection, segmentation 성능 측정이 가능하다.

3-3. LAG(Large-scale Attention based Glaucoma) Dataset



녹내장(glaucoma) 검출을 위한 데이터셋으로, 의료 진단 시스템 데이터셋이다.

3-4. STC(ShanghaiTech Campus) Dataset

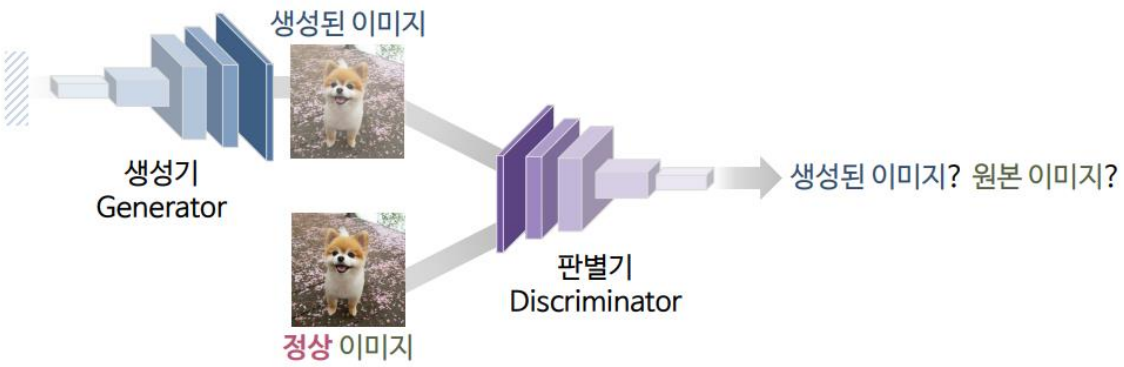
학교 캠퍼스의 CCTV 영상 데이터로서 27만 프레임의 학습 데이터와 4만 프레임의 테스트 데이터로 구성되어 있다. 데이터셋이 영상으로 이루어져 있으므로 Time-Series Image Dataset이다.

3. Anomaly Detection의 방식

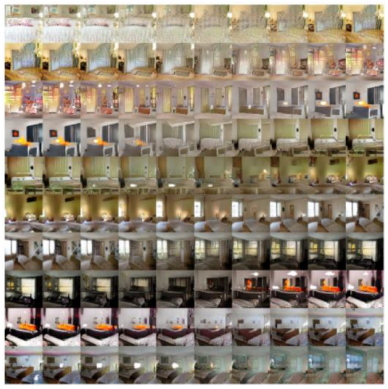
1. Reconstruction 방식 (AnoGAN)

AnoGAN은 GAN을 이용한 Anomaly Detection의 시초 버전이다. 네트워크는 DCGAN처럼 GAN의 구조에 CNN을 적용한 형태이다.

$$\min_G \max_D V(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}\left[\log D(\mathbf{x})\right] + \mathbb{E}_{\mathbf{z}\sim p_{\mathbf{z}}(\mathbf{z})}\left[\log(1 - D(G(\mathbf{z})))\right].$$



처음에는 정상 이미지만으로 GAN을 학습한다. GAN은 Latent Vector(z)로 이미지를 생성하는 Generator와 생성된 이미지와 원본 이미지를 판별하는 Discriminator로 구성돼있다.
(latent vector(z): 랜덤으로 뿌려진 벡터로 정규 분포에 따른 랜덤 값을 가짐, 이미지는 다차원 특징 공간의 한 점으로 표현이 가능한데 latent vector는 정규 분포를 따르는 공간에서 한 점이라는 의미임)
Generator로 생성된 이미지를 G(z), 원본 이미지를 x라고 할 때 Generator는 D(G(z))가 1이 되는 것을 원한다. 즉 판별기에 생성된 이미지를 넣었을 때 원본이라고 판단되기를 원한다. Discriminator는 D(x)는 1로, D(G(z))는 0으로 판별하는 것을 원한다. 즉 원본 이미지와 생성된 이미지를 잘 구분해내는 것을 원한다. 이것을 수식으로 말하면 Generator는 GAN Loss가 MIN이 되기를 원하고 Discriminator는 GAN Loss가 MAX가 되기를 원한다. 결국 두 적대적인 모델이 번갈아 학습을 하다가 D(G(z))가 1/2이 될 때, 즉 생성된 이미지가 원본인지 생성된 것인지 구분을 못하는 상태가 될 때 학습을 종료하는 방식이다.



훈련을 마치면 모델은 z에서 x로 가는 Generator를 학습한 상태가 된다. 그러나 반대의 경우인 x에서 z로 가는 매핑은 알지 못하는 상태이다. DCGAN에서 z를 조금씩 조정해 가면서 이미지를 생성하면, 이미지가 서서히 바뀐다고 말하는데 이것을 'walking in the latent space'라고 표현한다. 이것을 쉽게 말하면 z에 따라서 G(z)의 값이 변한다는 것이고, 최적의 z를 찾으면 x와 매우 유사한 이미지를 생성할 수도 있다는 것이다.

$$\mathcal{L}_R(\mathbf{z}_\gamma) = \sum |\mathbf{x} - G(\mathbf{z}_\gamma)|.$$

$$\mathcal{L}_D(\mathbf{z}_\gamma) = \sum |\mathbf{f}(\mathbf{x}) - \mathbf{f}(G(\mathbf{z}_\gamma))|,$$

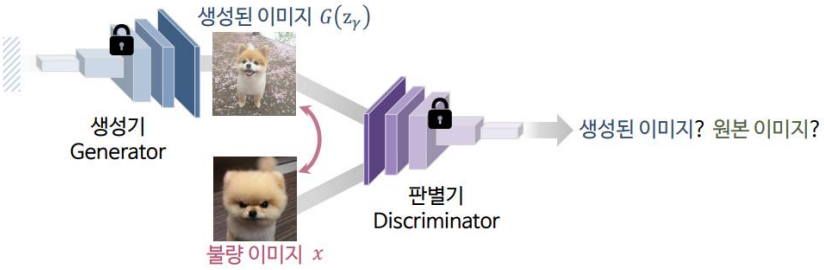
$$\mathcal{L}(\mathbf{z}_\gamma) = (1-\lambda)\cdot\mathcal{L}_R(\mathbf{z}_\gamma) + \lambda\cdot\mathcal{L}_D(\mathbf{z}_\gamma).$$

이후에는 생성기와 판별기를 모두 고정하고 Latent Vector인 z를 학습한다. 훈련을 할 때는 정상 데이터만 사용한다고 하였으니 결국 z는 정상 이미지의 잠재 공간(latent space)이 되도록 하는 것이다. 이러한 작업을 하기 위해서 Residual Loss와 Discrimination Loss가 제안되었다.
Residual Loss는 정상 이미지와 생성된 이미지의 L1 distance로 loss를 낮추면서 정상 이미지와 유사해지도록 학습하는 것이다.
Discrimination Loss는 정상 이미지의 판별 결과와 생성된 이미지의 판별 결과의 L1 distance로 loss를 낮추면서 판별기가 정상 이미지와 생성된 이미지를 구별하지 못하게 학습한다.

이 두 Loss를 linear combination한 것이 z를 학습하기 위한 Loss Function이다. 이 때 lambda값은 0.1이 가장 적합하다고 한다. (residual loss를 더 낮추는 것을 목표)

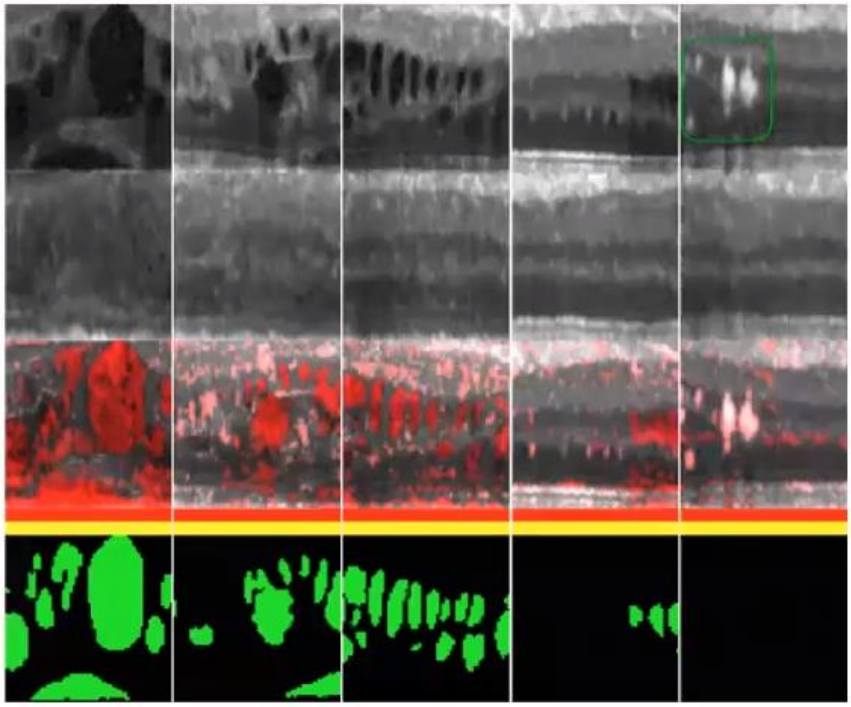
학습을 마친 z를 Generator에 넣으면 정상 이미지와 매우 유사한 가짜 이미지를 생성하기 때문에 이 가짜 이미지를 마치 정상인 것처럼 생각하고 비정상 이미지와 비교할 수 있게 된다.

$$Anomaly\ Score = (1-\lambda)\cdot\sum|x - G(z)| + \lambda\cdot\sum\left|f(x) - f\left(G(z)\right)\right|$$



Anomaly Detection을 할 때는 입력으로 비정상 데이터를 주고 Anomaly Score는 L(z)로 선정한다. 입력 이미지가 정상이면 당연히 score가 낮고 비정상이면 score가 높게 된다. 이 anomaly score가 임계치를 넘으면 anomaly라고 답할 수 있다. 논문에서는 Residual Image를 구하면 어떤 부분이 비정상인지도 확인할 수 있다고 한다.

Residual Image $X_r = |X-G(Z_R)|$ // 원본 이미지 – 생성된 이미지

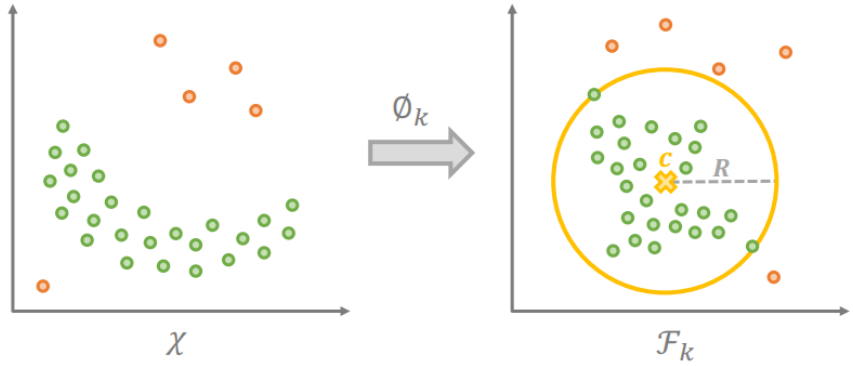


(원본 이미지, 생성된 이미지, 잔차 이미지, 실제 마스크(GT))

2. classification 방식 (Deep SVDD)

svdd란 feature space에서 정상 데이터를 둘러싸는 가장 작은 구(hypersphere)를 찾고, 해당 경계면을 기반으로 이상치를 탐지하는 방법이다. x라는 input space에 초록색에 해당하는 정상 데이터와 주황색에 해당하는 비정상 데이터가 있다고 가정하겠다. svdd는 먼저 input space에 있는 데이터들을 pi(k)라고 하는 kernel function을 이용해서 F_k라는 feature space로 매핑을 시킨다. 매핑 후 정상 데이터의 feature들을 잘 둘러싸는 가장 작은 구를 찾는 과정을 진행한다. 그리고 구 안 쪽

은 normal, 바깥쪽은 abnormal이라고 판단하는 것이다.



$$\min_{R,c,\xi_i} R^2 + \frac{1}{vn}\sum_i \xi_i$$

$$s.t. \parallel \phi_k(x_i) - c \parallel^2_{F_k} \leq R^2 + \xi_i,\ \xi \geq 0$$

목적함수는 위와 같다. Parameter는 c, R, slack이 있고 hyperparameter로는 phi, v가 있다.
(c: 구의 중심, R: 반지름, slack: penalty for soft margin, phi: kernel function, v: 반지름과 오차간의 trade-off)

첫 번째 항을 보면 svdd의 목적에 따라서 반지름을 줄이려고 하는 것을 볼 수 있다. 두 번째 항에서 slack은 softmargin에 해당하는(상대방 구역에 넘어가는, phi(x)와 중심 c사이의 거리가 R을 초과하는) instance에 대해 패널티를 부여한다.

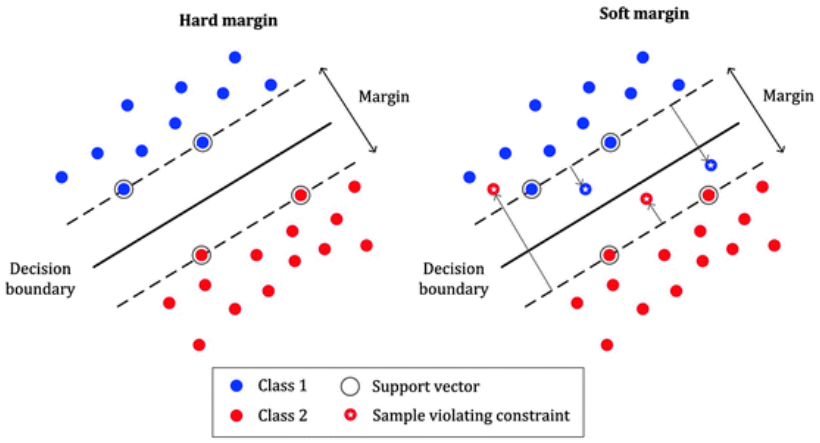
즉 패널티가 많을수록 오차가 많다고(정상 데이터가 비정상 구역에 있음, 원 밖에 있음)생각하면 되는데 hyper parameter인 v를 조정해서 오차를 더 집중해서 학습을 할 수 있다.

예를 들어 v가 작으면 두 번째 항이 더 커지므로 오차가 더 돋보인다. 따라서 오차를 막기 위해 구를 더 크게 만드는 경향이 있다. 반대로 v가 커지면 구가 더 작아지도록 학습이 된다. 즉 구가 커지면 오차가 줄어들고 구가 작아지면 오차가 늘어나는 trade-off 관계를 사용자가 v를 조정해서 적절하게 학습시키는 것이다.

즉 구를 최대한 작게 하면서 오차를 덜 발생시키려는 목적 함수라고 생각하면 된다.

이 때, [kernel function(phi)을 통해 mapping 된 input의 feature들]과 [구의 중심(c)]의 차이가 R^2에 slack을 더한 것보다는 작거나 같다고 정의한다.

(soft margin과 slack에 대한 이해는 아래 svm 설명을 참고한다.)

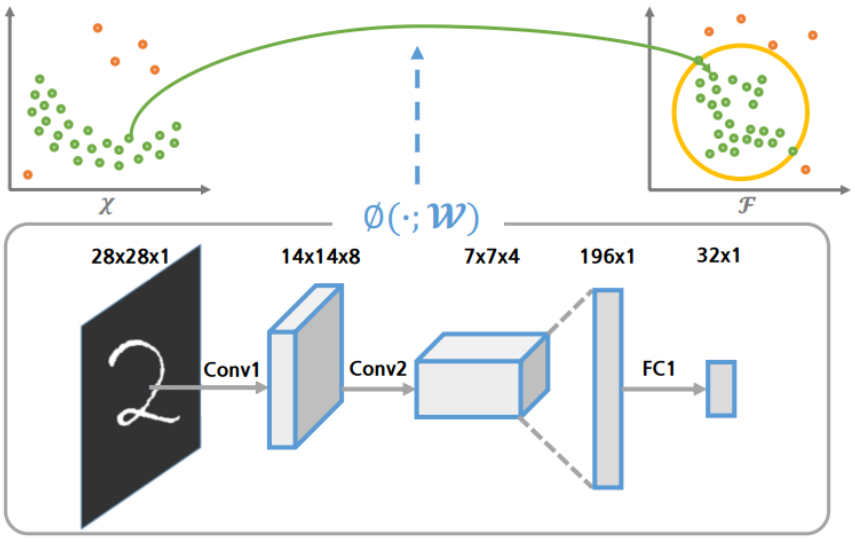


$$\frac{1}{2}\|w\|^2 + C\left(\sum_i \xi^{(i)}\right)$$

soft-margin은 soft margin svm에서 나오는 개념인데 상대방의 클래스의 마진 평면을 넘어가는 인스턴스를 허용하는 svm을 soft margin svm이라고 한다. soft margin 방법을 사용할 때는 자신의 마진 평면을 넘어가는 인스턴스에 대해서 패널티(slack)을 부여한다. 패널티는 자신이 속한 클래스의 마진 평면에서 떨어진 거리만큼 부여된다. 여기서 패널티는 오차에 해당이 되는데 이 오차를 줄이는 것을 목적으로 모델이 학습이 된다.

(svm에서는 오차를 줄이기 위해서 여백을 줄인다. 즉 hyperparameter C를 크게 하면 여백이 많이 줄어든다.)

Deep SVDD는 Kernel Function(phi) 부분을 Deep Learning으로 대체하여, 입력 데이터를 Mapping하는 Weight들을 학습시키겠다는 것이다.



$$\min_{R,W} R^2 + \frac{1}{vn}\sum_i \max\{0, \| \phi(x_i;W) - c \|^2 - R^2\} + \frac{\lambda}{2}\sum_l \| W^l \|_F^2$$

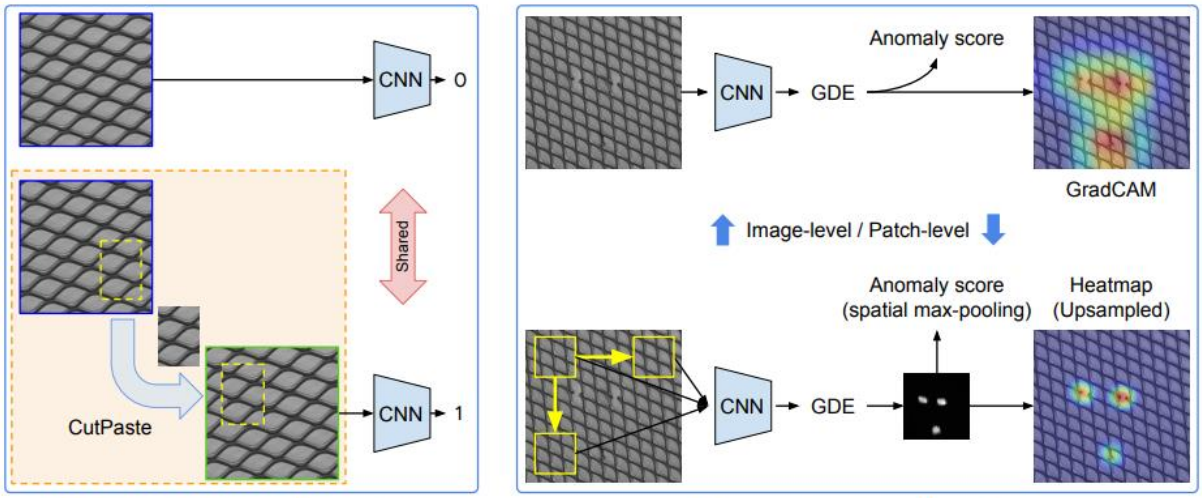
목적함수는 위와 같다. 먼저 구의 반지름을 최소화하는 것은 똑같고 feature와 중심 사이의 거리가 반지름보다 크면 패널티를 주고 작으면 패널티를 주지 않는 부분은 slack 처리와 비슷하다. 또한 오버피팅을 방지하기 위해 규제를 하는 항이 추가되었다. 이렇게 최적화를 하면 W는 각 데이터를 구의 중심 c(hyperparameter로 설정됨)에 가깝게 mapping하도록 학습된다.

$$s(x) = \| \phi(x; W) - c \|^2$$

anomaly score는 위와 같이 계산한다. 출력 feature에 대해 c에서 멀리 떨어진 정도를 anomaly score로 사용한다. 실제로 모델을 개발할 때는 AutoEncoder를 Pretrain해서 feature space(latent space)로 잘 mapping할 수 있는 weight를 구한다고 한다. 결국 DeepSVDD는 pretrain모델로 학습된 mapping weight들을 가지고 분류를 하는 classifier이다.

3. Feature matching 방식 (CutPaste)

cutpaste는 자르고 붙이는 방식의 augmentation을 사용하여 정상, 불량을 구분하도록 학습하는 자기 지도 학습 기반 anomaly detection이다.



(a) Learning Self-Supervised Representation (b) Anomaly Detection and Localization

cutpaste의 Augmenation 과정은 다음과 같다.

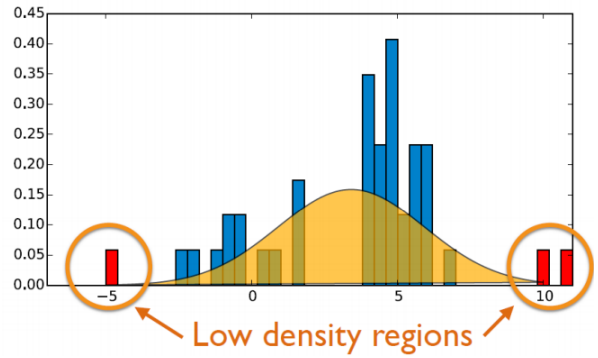
1. 정상 이미지의 작은 직사각형을 자른다.
2. 잘라낸 패치를 회전하시키거나 픽셀 값을 jitter한다. (jitter는 불규칙한 움직임이란 뜻이고 vision에서는 임의로 색상(Hue), 채도(Saturation), 명도(Lightness)를 변경하는 것을 말한다.)
3. 이 패치를 이미지의 랜덤 위치에 붙인다.

$$\mathcal{L}_{\mathrm{CP}} = \mathbb{E}_{x \in \mathcal{X}} \{ \mathbb{CE}(g(x), 0) + \mathbb{CE}(g(\mathrm{CP}(x)), 1) \}$$

(g:binary classifier, cp:cutpaste)

이후 원본인 정상 이미지는 0, CutPaste(CP)한 이미지는 1로 맞추도록 Cross Entropy loss를 이용해서 학습한다. 즉 라벨링된 정답 데이터가 없어도 직접 데이터를 생성해서 학습시키는 자기 지도 학습에 해당한다.

이후 학습된 모델에 실제 비정상 이미지를 입력하면 정상 여부와 함께 데이터의 특성 벡터를 확인할 수 있다. 추출된 특성 벡터들은 GDE(가우시안 밀도 추정법)를 통해 특성 벡터들의 분포를 추정하고, 분포의 극단점에 존재하는 특성 벡터들을 불량이라고 검출하게 된다. 즉 GDE를 통해 Anomaly Score를 구하는 방식이다.



(GDE: 데이터가 정규 분포를 따른다고 가정하고 실제 분포를 추정하여 분포의 바깥쪽에 위치한 경우에는 비정상으로 결정하는 이상치 탐지 알고리즘)

localization을 할 때는 GradCAM으로 정답에 가장 많은 영향을 미친 영역을 확인하면 된다고 한다.

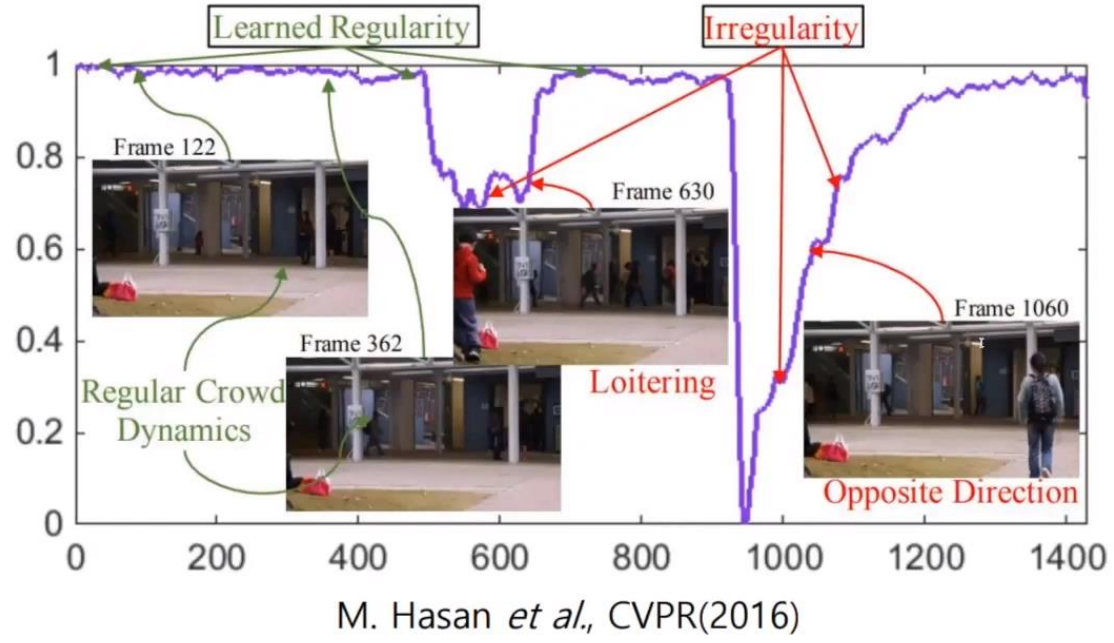
정리하면, CutPaste의 아이디어는 이상치가 전체가 다 이상하기 보다는 일부분만 이상한 경우가 대부분이라는 점을 고려해서 기존 이미지와 큰 차이가 없어 보이도록 하는 데이터 증강을 하고 자기 지도 학습을 통해 normal과 abnomal의 특징을 찾을 수 있다.

4. Video Anomaly Detection (Future Frame Prediction)

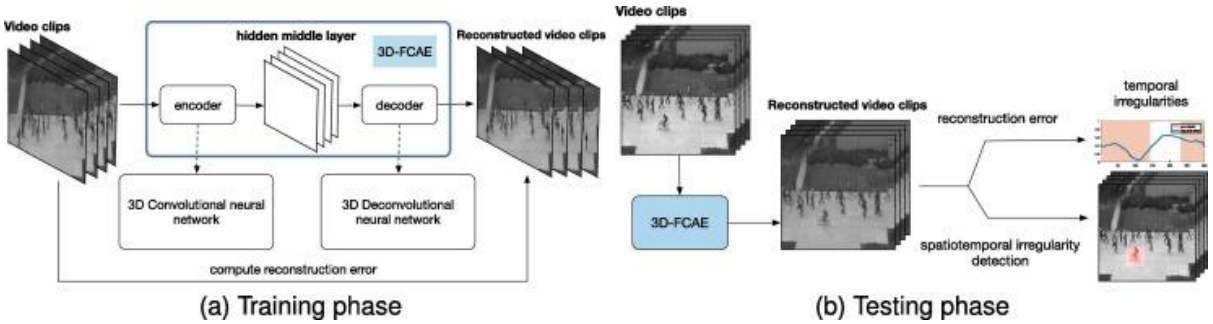
video anomaly detection은 비디오에서 예상하지 않은 상황을 검출하는 것을 목적으로 한다. 예를 들어 강도, 절도, 살인 등 예기치 않은 이벤트가 감시 카메라 상에서 발생하면 anomaly라고 탐지를 한다.

아래는 video anomaly detection의 한 예시이다. 정상 상황일 경우는 regularity score(정상일 확률를 나타내는 점수)가 높지만 비정상인 상황(사람이 밟지 말아야 할 잔디를 밟은 경우 등)에는

regularity score가 낮게 떨어진다. 따라서 video anomaly detection은 regularity score를 계산해서 일정 임계치 이하로 떨어지면 비정상이라고 분류할 수 있다.



대표적인 Video Anomaly Detection 방법으로는 3D CONV를 이용한 AutoEncoder이다.



학습을 할 때는 정상 상황만 복원하도록 하고 실제로 테스트를 할 때는 비정상 상황을 복원하라고 한다. 그러면 테스트할 때 reconstruction loss가 높기 때문에 이 경우에 regularity score를 낮추면 된다는 아이디어이다. 그런데 regularity score를 시간 순으로 나타내려면 당연히 시간 정보가 담긴 비디오를 학습해야 된다. 따라서 모델의 입력에는 비디오를 넣고 CONV연산도 3D CONV를 사용해서 영상의 위치 정보 뿐만 아니라 시간 정보를 학습할 수 있도록 한다.

그러나 AE의 성능이 좋아서(복원이 잘 돼서) 작은 비정상 변화를 감지하지 못하는 경우가 많았다고 한다.

따라서 새로운 Base Line인 Future Frame Prediction for Anomaly Detection이라는 방법이 나왔다.

AE처럼 Reconstruction을 하는 것이 아니라 Future Frame Prediction으로 Anomaly Detection을 하겠다는 것이다. 예를 들어 1~10번째 프레임을 입력으로 넣어서 11번째 프레임을 예측한 뒤 실제 11번째 프레임과 비교를 해서 오차가 크면 Anomaly라고 판정을 하는 방법이다. 물론 이 방식으

로 학습을 할 때도 정상 데이터로만 학습을 해서 비정상인 미래를 예측하지 못 하게 하는 것이다.

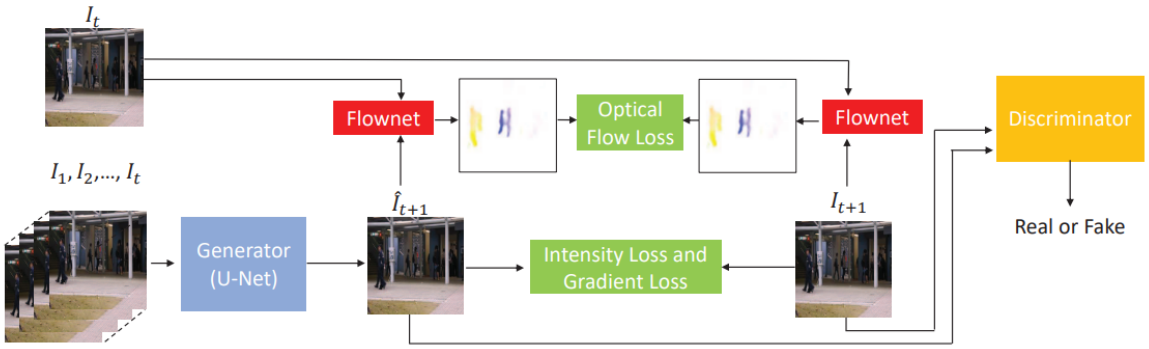


Figure 2. The pipeline of our video frame prediction network. Here we adopt U-Net as generator to predict next frame. To generate high quality image, we adopt the constraints in terms of appearance (intensity loss and gradient loss) and motion (optical flow loss). Here Flownet is a pretrained network used to calculate optical flow. We also leverage the adversarial training to discriminate whether the prediction is real or fake.

네트워크 구조를 보면 GAN방식을 그대로 따른다. Generator는 U-Net을 이용해서 future frame을 예측한다. Discriminator는 Patch GAN에 쓰이는 patch Discriminator를 사용해서 이미지의 특정 부분만 보고 생성된 future frame이 원본일 확률을 계산한다. 그러나 Generator와 Discriminator 구조만으로는 올바른 future frame을 보장할 수 없었다고 한다. 따라서 (현재 프레임, 미래 예측 프레임)간의 optical flow, (현재 프레임, 미래 실제 프레임)간의 optical flow를 계산해서 그 차이를 기반으로 모델을 학습을 했다고 한다. optical flow는 픽셀의 이동을 시각적으로 나타내는 방법(프레임과 프레임 사이에 변화가 있을 때 어떤 변화가 일어났는지 벡터로 표현하는 방법)이고 이전 프레임과 현재 프레임이 조금만 차이가 있어도 의미 있는 이동 정보가 나온다고 한다. 즉 작은 변화 감지에 용이한 optical flow도 같이 학습을 하면 예측 프레임과 실제 프레임의 작은 오차까지도 개선해서 future frame prediction 성능을 높일 수 있다고 한다.