지식 증류를 통한 자가 개선 BeatGAN

임에딘, 황채은

경희대학교 컴퓨터공학과

vlfdus0401@khu.ac.kr, hce9603@khu.ac.kr

Self-Improving BeatGAN via Knowledge Distillation

Edin Lim, Chaeeun Hwang

Computer Science and Engineering, Kyung Hee University

요 약

최근 사람들의 건강에 대한 관심사가 늘고 심장 질환에 대한 걱정이 커짐에 따라, 병원에서 부정맥을 진단하고 치료하는 것에 대한 중요도도 커지고 있다. 현재 병원에서는 환자들의 심전도 데이터를 측정 및 기록하고 분석하는 일을 전문의에게 의존하고 있으며, 이는 많은 수고로움을 동반한다. 따라서 본 연구에서는 그러한 문제점을 개선하고자 기존의 비지도 학습 기반 모델을 사용하여 비정상 심전도를 감지하는 BeatGAN에 지식 증류 기법을 도입하여 자가 개선을 하는 부정맥 예측 딥러닝 모델을 제안한다.

1. 서론

1.1. 연구배경

최근 기술과 의학이 발전함에 따라, 사람의 수명은 예전보다 많이 늘었다. 사람들은 늘어난 수명만큼 오랜 시간을 건강하게 보내려는 추세를 보이며, 이러한 관심은 당연히 자신과 가족의 건강 문제에 대한 관심으로 직결되고 있다.

최근 들어 많은 사람에게 자주 발생하고, 걱정을 끼치고 있는 질병 중에는 심장 및 혈관 질환이 있다. 심장 질환은 계속 늘어나는 추세이며, 그 중에서도 자주 언급되는 질환 중 하나가 바로 부정맥이다. 부정맥이란, 심장이 불규칙한 리듬이나 비정상적인 심박수를 갖는 것으로, 돌연사나 뇌졸중 등 생과 직결된 중질환을 유발할 수 있다. 텔레비전 매체와 인터넷을 통해 사람들은 다양하고 폭넓은 건강 지식과 질병에 대해 들을 수 있으며, 이에 따라 늘어나고 있는 부정맥에 대한 관심과 걱정도 커지게 되었다.

본 연구에서는, 정상 심전도 및 부정맥 심전도 그래프를 포함한 다양한 심전도 데이터를 수집, 분석하여 AI기반의 딥러닝 알고리즘을 통해 부정맥을 진찰해내는 것을 목표로 한다. BeatGAN은

생성자와 구분자 두 네트워크를 적대적으로 학습시키는 비지도 학습 기반의 생성 모델을 사용하여 주어진 심전도 데이터에서 비정상적인 beats를 감지한다. BeatGAN은 효율적으로 이상 심전도를 감지해낼 수 있으나, 추가적인 training 기법이 존재하지 않기 때문에 본 연구에서는 FRSKD(Feature Refinement via Self-Knowledge Distillation : 자기 지식 증류를 통한 특징 개선)을 BeatGAN에 결합하여 지식 증류 기법을 통해 자가 개선을 하는 딥러닝 모델을 연구한다[1,2].

1.2. 연구목표

비지도 학습 기반의 생성 모델을 이용하여 비정상적인 심전도를 감지할 수 있는 BeatGAN의 모델에 지식 증류 기법을 도입하여 자가 개선을 하는 부정맥 예측 딥러닝 모델을 구성하고, 기존의 모델보다 성능을 개선하여 더 높은 AUC를 얻는 것이 연구 목표이다.

2. 관련 연구

2.1. BeatGAN

적대적으로 생성된 시계열을 이용하여 비정상적인 리듬을 감지하는 알고리즘이다. (BeatGAN: Anomalous Rhythm Detection using Adversarially Generated Time Series [1]) BeatGAN은 다음과 같은 장점이 있다. 1) 비지도 방식(Unsupervised) : 레이블 없이 적용이 가능하다. 2) 효율성(Effectiveness) : ECG 데이터에서 거의 0.95 AUC의 정확도를 달성하고 매우 빠른 추론 속도(비트당 2.6ms)를 보인다. 3) 설명 가능성(Explainability): 비정상적인 패턴과 그와 관련된 시간 틱을 정확히 찾아내어 시각화 및 주의 집중을 위한 해석 가능한 출력을 제공한다. 4) 일반성(Generality) : ECG데이터 뿐만 아니라 다변수 모션 캡처 데이터베이스 (CMU Motion Capture Dataset)에서 비정상적인 움직임을 성공적으로 감지한다.

2.1.1. 이상 탐지 기술 (Anomaly Detection)

의도하지 않은 정상적인 시스템의 동작을 이상 현상이라고 한다. 이상 현상을 해결하기 위해서 발생한 현상이 '이상 상태'임을 인지해야 한다. 이상 현상이 정상의 양상과 많이 달라 쉽게 인지할 수 있을 때가 있지만, 정상과 굉장히 유사해서 인지하기 어려울 때도 있다. 이는 다양한 분야에서 발생할 수 있으며 이를 탐지하기 위해 '이상 탐지 기술' 연구가 진행되고 있다. 학습 시 비정상 샘플과 라벨의 유무에 따른 분류로 지도(Supervised) 이상 탐지와 비지도(Unsupervised) 이상 탐지가 있다. BeatGAN의 경우 비지도 이상 탐지 기술로 라벨링의 과정이 필요가 없는 장점이 있지만 다른 방법 대비 정확도가 낮은 단점이 있다.

2.1.2. 재구성 기반 판별 방식 (Reconstruction)

이상 판별할 데이터를 저차원 형태의 잠재 구조(latent structure)를 획득하고 인위적으로 재구성한 데이터를 생성하기 위한 모델을 사용한다. 대표적인 방법으로 PCA(Principal Component Analysis, 주성분 분석)이 있다. 주어진 데이터에 대해 PCA를 이용하면 데이터의 차원을 축소하고 복원하는 과정을 통해 비정상 샘플을 검출한다. 단점으로는 선형 재구성으로만 제한이 된다. 딥러닝 기반 기술에서는 주로 오토인코더(Autoencoder)기반의 방법론이 자주 사용된다. AE(Auto-Encoder), VAE(Variational Auto-Encoder), LSTM 기반 인코더-디코더 구조가 있으며, PCA와 달리 비선형 차원 축소를 다룰 수 있다.

2.1.3. GAN (Generative Adversarial Networks)

생성자(Generator)와 구분자(Discriminator) 두 네트워크를 적대적(Adversarial)으로 학습시키는 비지도 학습 기반의 생성모델(Unsupervised generative model)이다. GAN은 생성자가 만든 가짜 데이터가 진짜 데이터와 비슷하여 판별자가 진위를 판별하지 못할 때까지 알고리즘을 개선하는 방식으로 학습을 진행한다. 이처럼 GAN으로 학습하는 생성자는 진짜 같은 가짜 데이터를 만들어내기 때문에, 유명 화가의 화풍을 입힌 이미지나 음성 변조 파일, 영상 등 다양한 콘텐츠 분야에서 활용되고 있다.

2.2. Feature Refinement via Self-Knowledge Distillation

2.2.1 지식 증류 (Knowledge Distillation)

딥러닝에서 지식 증류는 큰 모델(Teacher Network)로부터 증류한 지식을 작은 모델(Student Network)로 전달하는 과정이다. 복잡한 모델은 실제 서비스로 배포할 때 사용자들에게 적합하지 않을 수 있다. 만약 작은 모델이 더 큰 모델만큼의 성능이 나온다면 배포 시 적합하며 컴퓨팅 자원 측면에서도 효율적일 것이다. 지식 증류는 학습 과정에서 큰 네트워크로부터 증류된 지식이 작은 네트워크로 전달하고 그의 성능을 높이는 것에 목적이 있다.

모델이 이미지 클래스를 분류할 때 각 클래스의 확률값이 출력된다. 가장 높은 확률을 보이는 클래스에 따라 예측을 하는 구조로, 교사 네트워크의 분류 결과를 학생 네트워크의 분류 결과와 비교시켜서 학생이 교사를 모방하도록 학습시킨다. 여기에서 예측한 클래스 이외의 확률값에 주목하여 학생 모델이 이러한 정보의 손실 없이 학습할 수 있도록 기존 softmax 함수에 하이퍼파라미터 T(Temperature)를 반영한다. 이를 사용하면 낮은 입력값의 출력을 크게 만들어주고, 큰 입력값의 출력은 작게 만들어 전체적으로 출력값을 부드럽게 만들어준다. 이러한 방법으로 지식 증류는 교사와 학생 두 네트워크의 분류 결과를 hard label이 아닌 소프트 레이블(soft label)로 사용하여 학생 네트워크가 학생 네트워크를 모방하여 학습할 때의 이점을 최대화하고 성능을 높인다.

2.2.2 자가 지식 증류 (Self-Knowledge Distillation)

사전에 학습된 교사 네트워크의 출력값에 softmax 처리한 값을 활용하여 학생 네트워크를 학습시키는 기존의 지식 증류 방법과 달리, 자가 지식 증류는 단일 모델 내부에서 지식을 증류하는 방법이다. 자가 지식 증류는 데이터 증강 기반 방식과 보조 네트워크 기반 방식으로 크게 두가지가 있다. 데이터 증강 기반 방식은 딥러닝 모델을 학습시키는 데 필요한 데이터를 확보하는 기법 중 하나로, 소량의 학습 데이터에 인위적인 변화를 가하여 새로운 학습 데이터를 확보하는 방법이다. 단점으로 데이터의 증강 과정에서 지역 정보(local information)을 잃어 지식 증류의 성능을 향상시킬 수 있는 기술인 특징 증류(feature distillation)을 활용하기 어렵다. 보조 네트워크 기반 방식은 분류기 네트워크 중간에 추가 경로를 활용하고 지식 전달을 통해 추가 분기를 유도하여 유사한 출력을 생성한다. 하지만 이 방식은 특징을 더욱 복잡하게 만들 수 없는 단점이 있다.

2.2.3. FRSKD (Feature Refinement via Self-Knowledge Distillation)

기존의 자가 지식 증류 방법의 단점을 보완하여 개선한 방법이 FRSKD이다. FRSKD는 자가 지식 증류를 위해 특징 증류를 활용할 수 있고 소프트 레이블로 정제된 지식을 생성하여 사용이 가능하다. 따라서 지역 정보의 보존을 강조하는 분류 및 의미 분할에 적용이 될 수 있으며 다른 자가 지식 증류 방법과도 호환이 되는 장점이 있다. BiFPN의 구조를 수정하여 하향식과 상향식 경로를 적용한 자가교사 네트워크를 모델링한다. 특징 증류(feature distillation)는 주의 전달(attention transfer)을 적용했으며 이를 통해 분류기 네트워크가 자가 교사 네트워크에서 정제된 기능맵의 위치를 학습하도록 유도한다. 최적화 목표는 Cross Entropy 손실함수(L_{CE})와 특징 증류의 손실함수(L_f), 분류기 네트워크와 자가 교사 네트워크 각 출력값에 대한 손실함수(L_{KD}) 총 세가지를 통합하여 L_{FRSKD}로 구성한다.

3. 프로젝트 내용

3.1. 시나리오

BeatGAN은 대규모의 리듬 시계열 데이터, 즉 ECG(심전도) 데이터에서 효과적이고 효율적인 방법으로 비정상적인 심전도를 감지해낼 수 있는 알고리즘이며, FRSKD는 지식 증류 방법 중 기존의 자가 지식 증류 방법의 단점을 보완한 방법으로 보조 자가 네트워크 교사를 활용하여 분류기 네트워크에 정제된 지식을 전달함으로써 데이터 증강 및 모델의 학습에 효과적인 알고리즘이다.

BeatGAN은 적대적 생성 접근 방식을 사용한 재구성 오류의 정규화와 시계열 워핑을 사용한 데이터 보강으로 견고성이 보장되어 있고, 직관적인 접근 방식을 제공하지만 데이터를 증강시키는 것은 따로 언급되어 있지 않다. 따라서 본 연구에서는 BeatGAN의 알고리즘 성능을 좀 더 효율적으로 극대화 시키기 위해, FRSKD의 자가 지식 증류 방법을 접목하여 BeatGAN에 제공되는 데이터를 효과적으로

증강시킨다. BeatGAN의 기존 데이터를 FRSKD를 통해 데이터를 증강 및 학습하고, 이렇게 정제된 데이터를 다시 BeatGAN의 데이터에 보강함으로써 결과적으로 BeatGAN의 성능을 향상시키고 더 높은 AUC를 끌어낼 수 있도록 한다.

BeatGAN과 FRSKD는 둘 다 성능이 보장되어 있는 알고리즘이며 공개된 코드로 배포되고 있으므로, 이 공개된 코드를 취합하여 효율적인 알고리즘을 도출한다.

3.2. 요구사항

기존의 모델인 BeatGAN은 ECG 데이터에서 비정상적인 샘플을 감지할 때 우수한 성능을 보여준다. 이 모델은 대규모의 리듬 시계열 데이터에서 비정상적인 패턴과 그와 관련된 시간 틱을 정확히 찾아내며, 거의 0.95 AUC의 정확도를 달성하고, 매우 빠른 추론 속도(비트당 2.6ms)을 보여준다. 하지만 기존의 BeatGAN 모델은 학습하는 데이터의 증강 부분이 부족하므로, 본 연구에서는 FRSKD의 지식 증류 기법을 접목하여 BeatGAN의 데이터를 증강하는 데에 목표를 둔다. 따라서, 본 연구에서는 FRSKD의 지식 증류 알고리즘을 BeatGAN의 데이터 증강에 접목하여 기존의 모델보다 더 높은 AUC 정확도를 보여줄 수 있어야 한다.

3.3. 시스템 설계

3.3.1 BeatGA 네트워크 구조

BeatGAN 모델의 네트워크 구조는 그림 1에 묘사되어 있다. 오토인코더는 인코더G_E와 디코더G_D로 구성되어 있으며, 입력 데이터 x가 인코더로 들어오면 은닉 벡터 z로 압축되고 $G_D(z)$는 x'를 생성한다. 여기에 적대적인 학습 방식인 GAN 기술을 추가하여 Discriminator(D)가 정규화의 역할을 수행한다.

디코더 네트워크인 G_D의 구조는 DCGAN의 생성자 구조와 비슷하다[5]. DCGAN은 기존 GAN에 존재했던 완전 연결된 구조의 대부분을 CNN구조로 대체한 것이다. G_D는 그림 2와 같이 1D transposed 컨볼루션 레이어들과 Batch-norm, ReLU로 구성되어 있다. G_E 는 그림 3과 같이 G_D와 거의 유사하며 반대의 방향으로 되어있다. G_E는 1D 컨볼루션 레이어들과 Batch-norm, Leaky ReLU로 구성되어 있다.

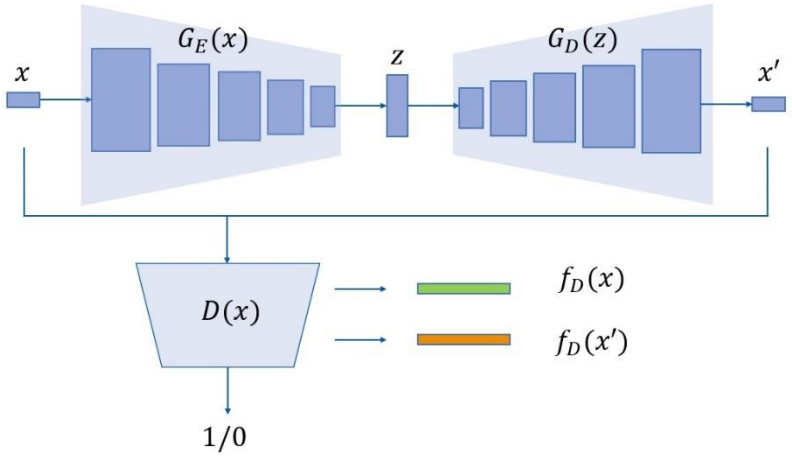


그림 1. BeatGAN 네트워크 구조

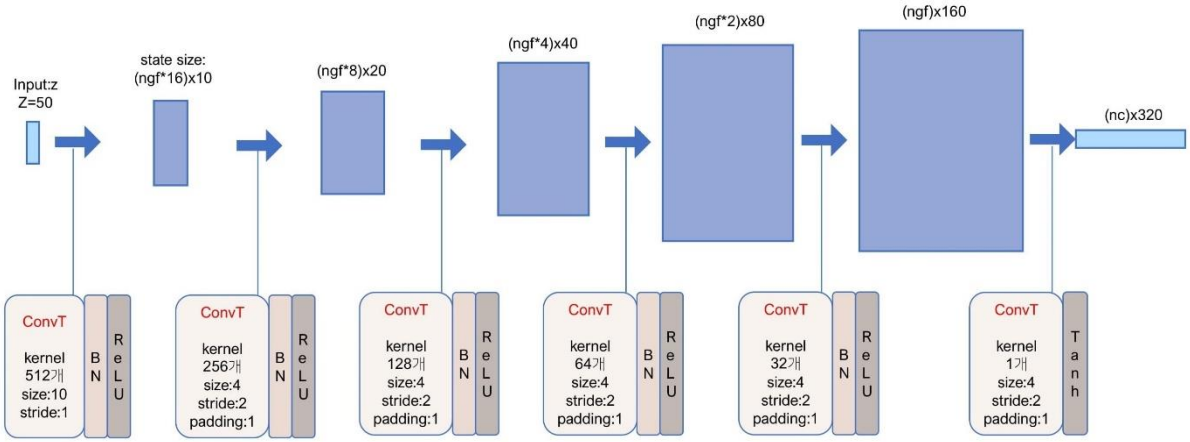


그림 2. 디코더 G_D구조

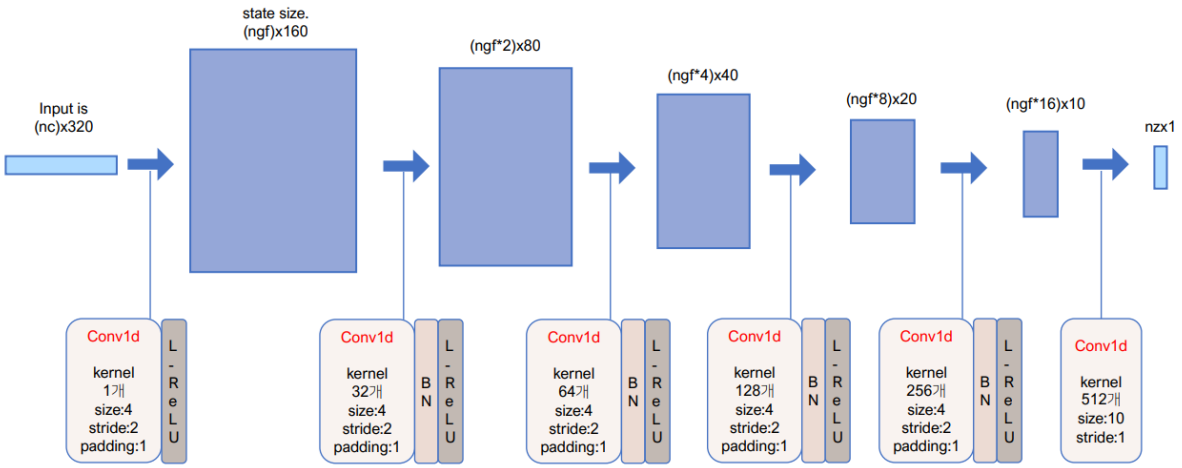


그림 3. 인코더 G_E구조

3.3.2 최적화 목적

우리는 지식 증류 기법을 활용하기 위해 교사 네트워크와 학생 네트워크를 설정했다. 기존 BeatGAN 모델을 사전에 훈련시키고 이를 교사 네트워크로 삼아서, 데이터에 대해 예측한 결과를 학생 네트워크가 전달받는다. 학생 네트워크는 훈련하는 과정에서 (1) 적대적 정규화를 통한 재구성의 손실, (2)학생이 예측한 결과와 교사로부터 전달받은 결과와의 손실을 통합하여 업데이트를 진행한다.

기존의 BeatGAN에서 재구성 모델의 손실 함수는 다음과 같으며 이를 최소화하는 것이 목적이다.

$$L_G = ||x - x'||_2 + \lambda ||f_D(x) - f_D(x')||_2 \quad (1)$$

왼쪽 항은 G를 통해 생성된 x'와 x의 복원 오차(reconstruction error)이다. 이는 이상 점수(anomaly score)가 되어 threshold와 비교를 통해 이상 여부를 결정한다. 오른쪽 항은 특징 일치 손실(feature matching loss)로, 손실을 계산하기 위해 D의 은닉층의 활성화 벡터 $f_D(\cdot)$을 이용하여 생성된 x'와 x의 특징 사이가 유사한지를 고려한다.

다른 지식 증류 기법과 유사하게 본 연구에서도 소프트 레이블을 통해 증류를 수행한다. 학생 네트워크가 출력하는 예측 결과와 교사 네트워크가 예측하는 결과에 대해 쿨백-라이블러 발산(KL divergence)을 통해 다음과 같이 손실 함수를 정의한다.

$$L_{KD}(x; \theta_s, \theta_t, T) \quad (2)$$
$$= D_{KL}(softmax(\frac{f_s(x; \theta_s)}{T}) || softmax(\frac{f_t(x; \theta_t)}{T}))$$

T는 온도 매개변수, s와 t는 각각 학생과 교사 모델 매개변수이다. 우리는 위의 손실함수를 통합하여 다음과 같은 최적화 목적을 구성했다.

$$L = L_G + L_{KD}(x; \theta_s, \theta_t, T) \quad (3)$$

자가 개선 BeatGAN의 구조를 손실함수와 통합하여 그렸을 때 그림 4와 같이 나타낼 수 있다.

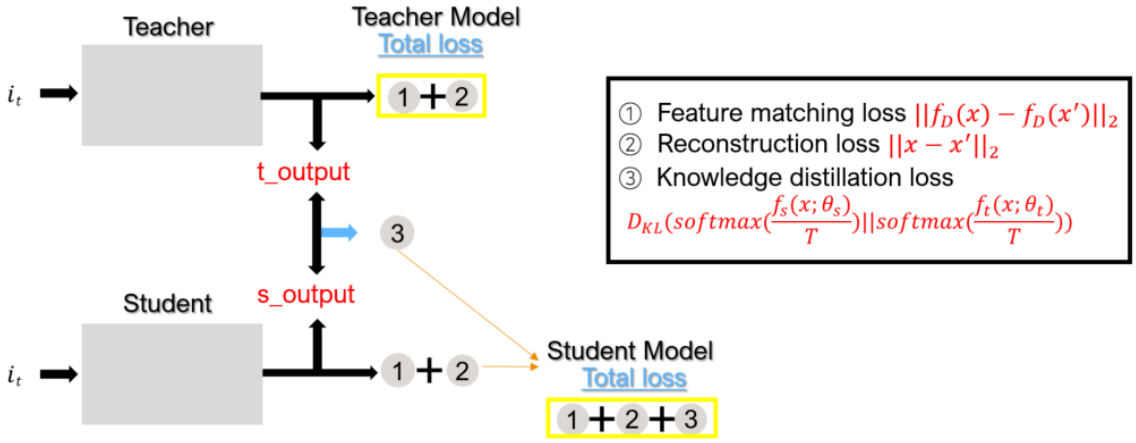


그림 4. 자가 개선 BeatGAN의 손실함수

3.3.3 자가 개선 BeatGAN 네트워크 구조

이번 연구에서 제안하는 자가 개선 BeatGAN (Self-Improving BeatGAN, SI-BeatGAN)의 구조는 그림 5에 묘사되어 있다. Teacher model은 사전에 훈련이 되어 있고 가장 높은 성능을 보이는 입력 데이터 x가 들어오면 각 model에서는 Generator을 통해 x'가 생성된다. Generator을 통해 생성된 x'와 x의 복원 오차(reconstruction loss), Discriminator의 은닉층의 활성화 벡터에 의해 x'와 x의 특징 사이를 고려하는 특징 일치 손실(feature matching loss), 그리고 student model이 예측한 결과와 teacher model로부터 전달받은 결과와의 손실을 통합하여 업데이트를 진행한다.

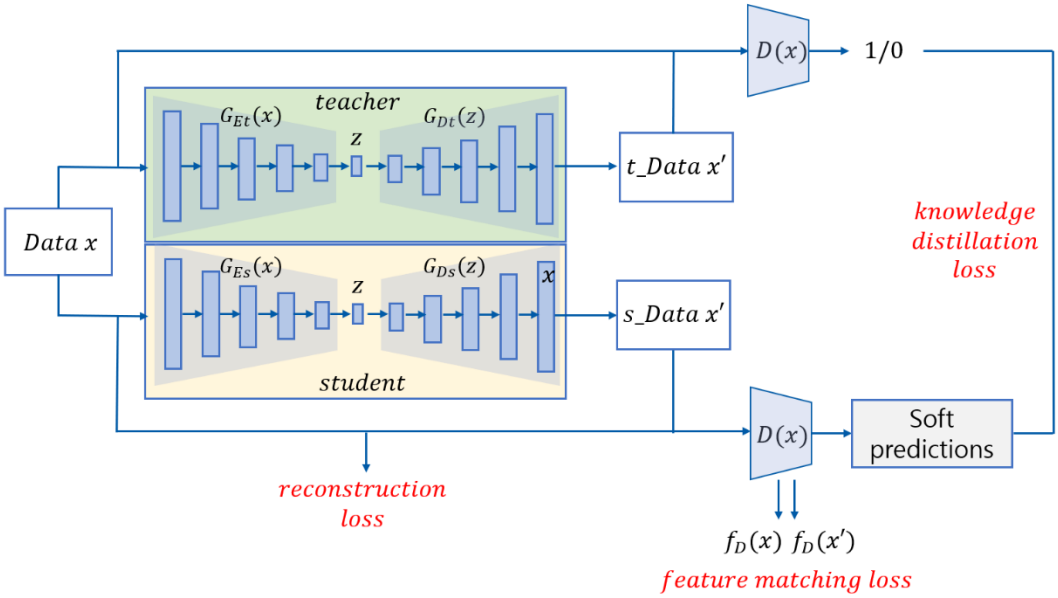
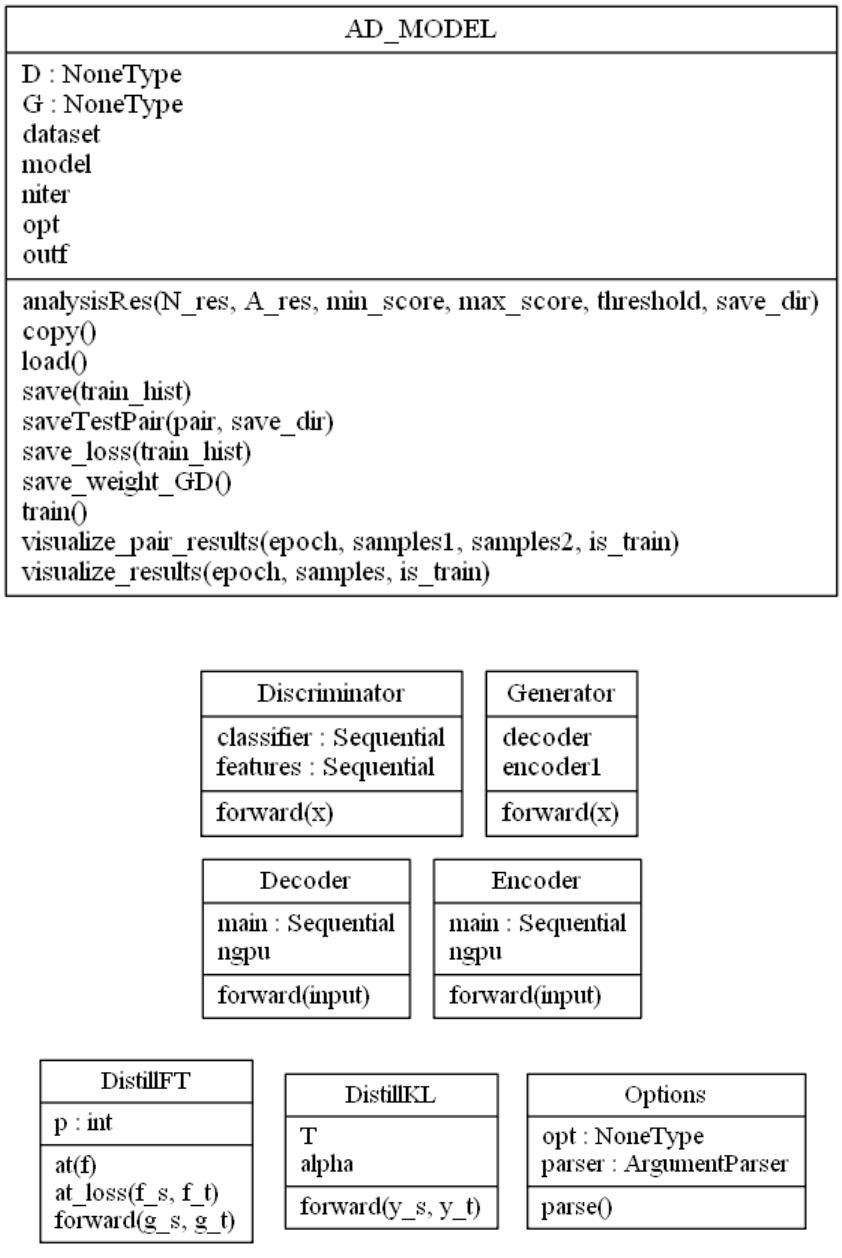


그림 5. 자가 개선 BeatGAN 구조

3.3.3 UML Diagram

3.3.3.1 Class Diagram

제시한 모델의 클래스 다이어그램은 다음과 같다(그림 7). 클래스는 크게 BeatGAN_t, BeatGAN_s, AD_MODEL, Decoder, Encoder, Discriminator, Generator, DistillKL, DistillFT, Options가 있다.



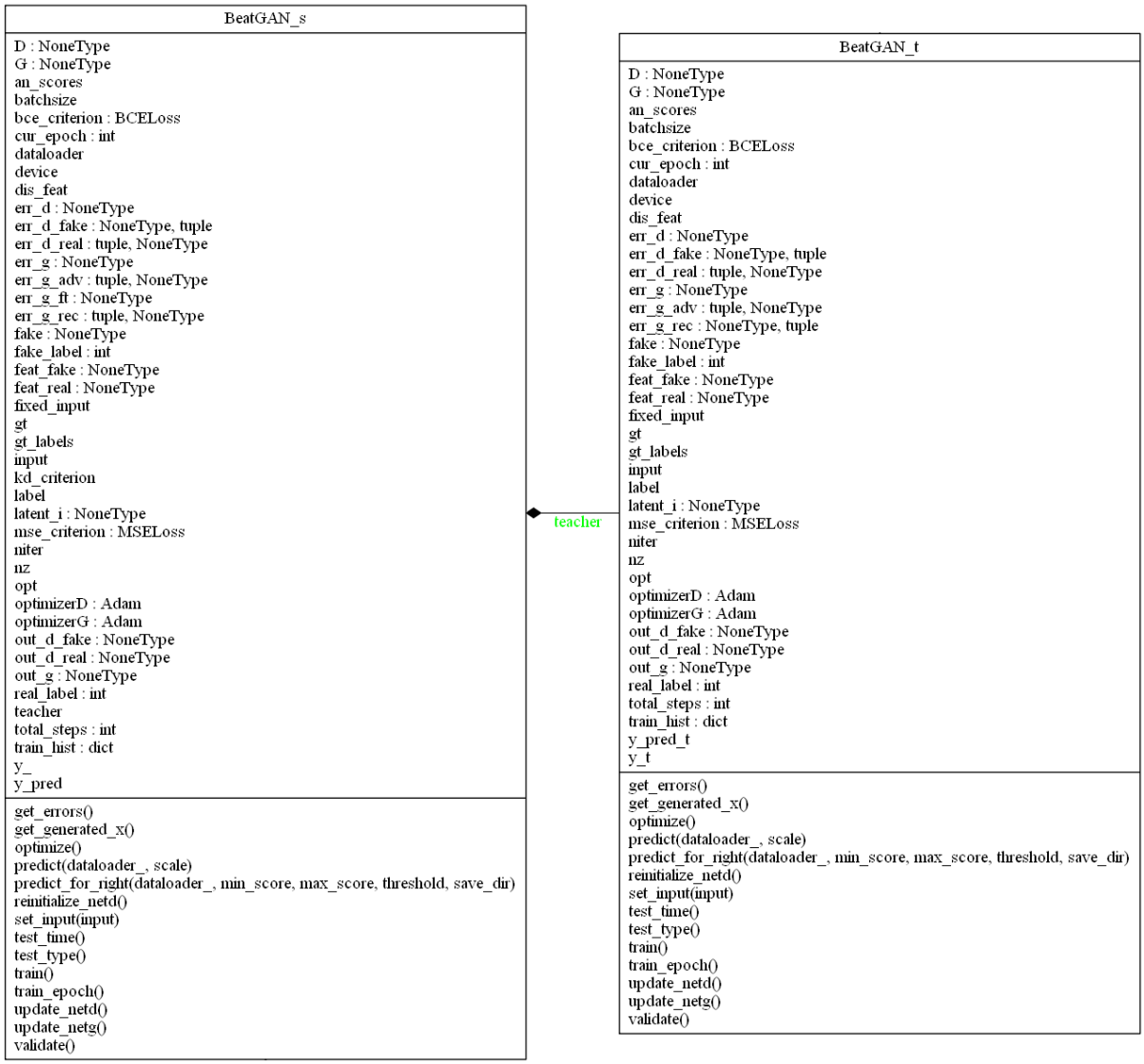


그림 7. Class Diagram

3.3.3.2 Package

우리가 제안하는 모델의 패키지들은 아래의 그림8과 같다.

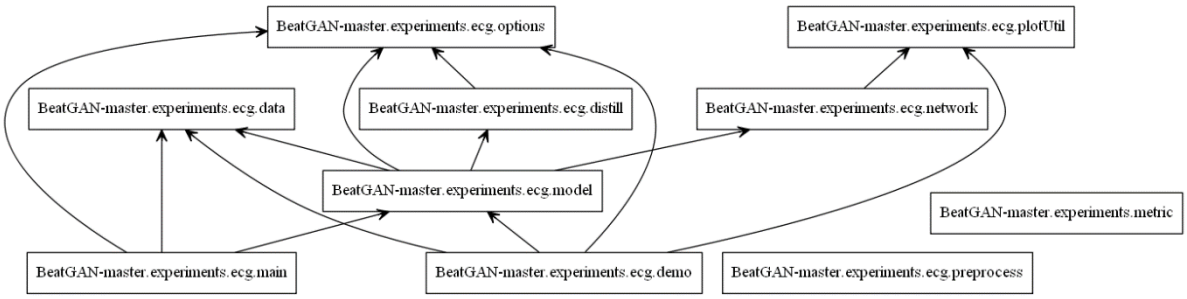


그림 8. Packages

3.3.4 주요 구성

3.3.4.1 데이터 정규화 [data.py]

- seed는 42로 고정되어 있다.
- normalize() 메소드는 입력 데이터에 대해 정규화를 한다. 모든 데이터의 포인터가 동일한 정도의 스케일로 반영되도록 해주는 것이 목표이다. 정규화는 [-1,1]범위에서 이루어지며, 입력 파라미터를 x라 하면 리턴되는 값 x'은 다음과 같다.

$$x' = 2\frac{x - min\ x}{max\ x - min\ x} - 1$$

- Dataloader은 batch 기반의 딥러닝 모델 학습을 위해 mini batch를 만들어준다. 학습 도중 CPU의 작업을 몇 개의 코어로 사용해서 진행할지 설정해주는 num_worker은 해당 연구의 진행이 로컬 PC에서 이루어지기 때문에 default(=0)으로 설정하였다.

3.3.4.2 AD_MODEL, Encoder, Decoder으로 구성된 [network.py]

- AD_MODEL

Student model과 Teacher model은 train을 하면서 가장 좋은 성능을 보이는 결과에 대해 save_weight_GD() 메소드를 통해 모델의 Generator와 Discriminator를 업데이트를 한다. 사진은 train할 때 콘솔에서 보이는 과정으로, 현재 99번째 epoch을 돌아 AUC가 약 0.9106이고, 지금까지 epoch을 돌면서 가장 성능이 좋았던 것은 AUC가 약 0.9220인 85번째 epoch임을 알 수 있다. 해당 모델은 85번째 epoch에서 훈련을 마치고 Generator과 Discriminator이 업데이트가 되어있는 상태이다. 이때 Generator과 Discriminator은 save_weight_GD() 메소드에 의해 "모델명"+"_folder_0"+"_D.pkl와 "모델명"+"_folder_0"+"_G.pkl 로 저장된다. Teacher model명은 실행파일에서 beatgan, Student model명은 beatgan_s로 지정되어 있다. 아래의 사진은 student model이 train을 할 때 저장되는 pkl파일들이다. train을 시작할 때, load()와 copy() 메소드를 통해 사전에 훈련된 Teacher의 D와 G

파일을 불러와서 저장한다. 이때 teacher 모델은 deep-copy가 되었기 때문에 student model이 훈련하는 동안 파일이 수정되지 않는다.

```
Epoch: [99] [ 100/ 975] D_loss(R/F): 0.000102/0.000303, G_loss: 0.171564
Epoch: [99] [ 200/ 975] D_loss(R/F): 0.000110/0.000054, G_loss: 0.190369
Epoch: [99] [ 300/ 975] D_loss(R/F): 0.000077/0.000155, G_loss: 0.170448
Epoch: [99] [ 400/ 975] D_loss(R/F): 0.000598/0.000006, G_loss: 0.205931
Epoch: [99] [ 500/ 975] D_loss(R/F): 0.000264/0.000070, G_loss: 0.162630
Epoch: [99] [ 600/ 975] D_loss(R/F): 0.000176/0.000365, G_loss: 0.169887
Epoch: [99] [ 700/ 975] D_loss(R/F): 0.000051/0.000545, G_loss: 0.144301
Epoch: [99] [ 800/ 975] D_loss(R/F): 0.001325/0.000022, G_loss: 0.138671
Epoch: [99] [ 900/ 975] D_loss(R/F): 0.000440/0.000017, G_loss: 0.150050
[99] auc:0.9106 th:0.0299 f1:0.6873      best_auc:0.9220 in epoch[85]
```

그림 9 . model의 train : epoch 99번째 진행하는 모습

그림 10. pkl파일 (load, copy, save)

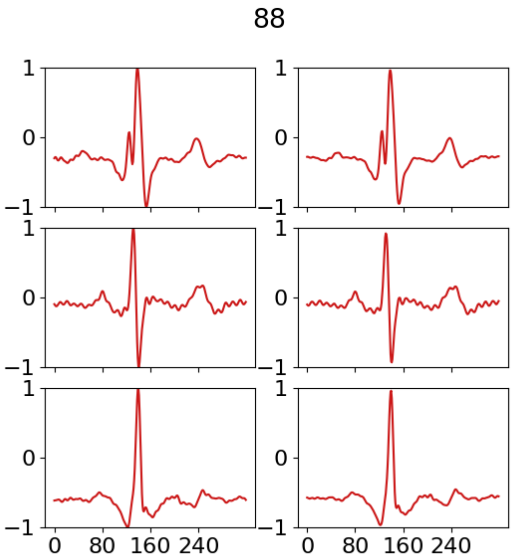


그림 11. visualize_pair_results()메소드로 생성된 88번째 epoch의 그림

- Encoder, Decoder

Encoder과 Decoder의 자세한 구조는 그림2, 그림 3에 나와있다.

3.3.4.3 Discriminator, Generator, Teacher model, Student model으로 구성된 [model.py]

model.py에는 Discriminator, Generator, BeatGAN_t, BeatGAN_s 클래스로 구성되어 있다.

- Discriminator
  - 실제 데이터가 주어졌을 때 높은 확률을 리턴하고, 가짜 데이터가 주어졌을 때 낮은 확률을 리턴하는 방향으로 weight를 업데이트하며, 진짜 데이터를 잘 가려내는 것이 목표이다.
  - 인코더에서 리스트 형태의 레이어들을 받아온다. Features는 가장 마지막에 있는 레이어를 제외한 모든 레이어이고 classifier은 가장 마지막 레이어이다. 마지막 레이어에서는 시그모이드를 이용한다.
- Generator
  - D가 진짜인지 구별할 수 없을 만큼 진짜 같은 가짜 데이터를 생성한다. Latent vector z가 주어졌을 때 G를 통하여 가짜 데이터G(z)를 생성하고 이를 D의 입력으로 주어지면 D(G(z))는 높은 확률을 리턴해주는 방향으로 weight를 업데이트한다. 마치 실제 데이터인 것처럼 진짜스러운 데이터를 만드는 것이 목표이기 때문이다.
  - 오토인코더 구조로 되어 있으며, Encoder-Decoder 형태로 이뤄져 있다. 입력 데이터 x가 인코더로 들어오면 생성되는 데이터를 latent_i로, latent_i가 디코더로 들어가면 출력 데이터를 gen_x로 받으며 두 변수를 리턴한다.
- Teacher model (BeatGAN_t)
  - BeatGAN_t는 기존의 BeatGAN 모델과 구성이 같다.
  - teacher model, student model 모두 Adam 옵티마이저를 사용하며, 이는 관성을 이용해 국소 최적해 문제를 해결하고 각 학습 파라미터마다 다른 학습률을 적용한다. 많은 작업에 대해 대체로 SGD 옵티마이저보다 더 나은 성능을 보여준다.
  - Discriminator, Generator은 총 100개의 epoch을 돌며 train과정을 거친다. update_netg()메소드는 Generator 네트워크를 업데이트하며, err_g_adv(복원오차), err_g_rec(특징일치손실)을 각 MSE로 구해서 Generator의 error인 err_g에 더하여 역전파시킨다. update_netd()메소드는 Discriminator 네트워크를 업데이트하며, err_d_real은 out_d_real와 real_label(=1)의 BCE loss이고, err_d_fake는 out_d_fake가 fake_label(=0)의 BCE loss이다. 이후 둘을 합쳐서 Discriminator의 error인 err_d를 역전파시킨다. 이렇게 얻은 error들은 get_errors()메소드를 통해 값들이 저장된다.
  - predict() 메소드에서는 리턴되는 값이 y_, y_pred이다. y_는 gt_label(ground truth label), y_pred는 an_scores(error)를 받아온다.
- Student model (BeatGAN_s)

- BeatGAN_s는 훈련하는 과정에서 BeatGAN_t로부터 지식을 증류시켜 전달받고 모방 학습에서 성능을 높인다.
- teacher model과 크게 다른 점은, update_netg()메소드에서 generator의 error인 err_g에 복원오차와 특징일치손실에 지식 증류 손실을 추가로 더한다. predict() 메소드를 통해 teacher model의 y_와 student model의 y_pred를 받아와 kd_criterion으로 지식 증류 손실을 계산한다.

3.3.4.4 지식을 증류하는 [distill.py]

distill.py에는 DistillKL과 DistillFT 클래스가 있다.

- DistillKL

  Knowledge Distillation Loss를 계산하는 클래스로, 해당 식은 3.3.3.(2)에 기재되어 있다. T는 온도 매개변수이고 alpha는 loss앞에 곱해지는 매개변수로 사용자가 두 변수를 조정할 수 있다. 해당 클래스의 forward함수에서 학생 네트워크가 출력하는 예측 결과를 log_softmax하고, 교사 네트워크가 예측하는 결과를 softmax하여 쿨백-라이블러 발산을 통해 loss를 계산한다. 마지막엔 loss를 리턴한다.

- DistillFT

  Feature Distillation Loss를 계산하는 클래스로, 본 연구를 시행착오하는 과정에서 특징증류손실을 계산하는 손실함수을 만들고 함수를 구현했다. FRSKD에서는 classifier 네트워크가 teacher 네트워크로부터 세련된 기능 맵의 위치를 모방하도록 특징증류를 추가했었으며 우리는 이 함수에 student 네트워크와 teacher 네트워크를 넣어 손실을 리턴받게 하였다. 손실함수는 다음과 같다.

$$L_F(Fs, Ft; \theta_s, \theta_t) = \Sigma_{i=1}^{n} ||\phi(Fs_i) - \phi(Ft_i)||_2$$

ϕ 는 L2정규화를 하여 channel-wise pooling를 하는 매개변수이다. Fs과 Ft는 각각 student와 teacher의 feature이고 θ_s, θ_t는 각각 student network와 teacher network이다.

4. 프로젝트 결과

4.1 개발 환경

프로젝트 진행은 로컬 컴퓨터로 진행하였다. 기존에는 코드 서버를 이용해서 rtx3090을 빌리고 연구를 하기로 했으나 가끔 무한 로딩이 되는 에러가 걸릴 때가 있고 가상환경 세팅, cuda, cudnn, pytorch 설치로 세팅을 전부 해놓아도 cuda 인식이 제대로 되지 않았다. 코드 서버의 Ubuntu 18.04.3 LTS bionic x86_64에서 계속 진행하다가 결국 로컬 세팅으로 바꾸었다. 개발 환경은 다음과 같다.

- OS : Window10
- VGA : GTX1660super
- VGA Driver : 472.12
- CUDA ver :11.0
- CUDNN ver : 8.0.5
- Anaconda : Anaconda3-2021.05-Windows-x86_64.exe
- Python ver : 3.7.6
- PyTorch ver : 1.7.1
- VS Code (Terminal : Git Bash)

4.2 연구 데이터

데이터는 MIT-BIH arrhythmia database를 이용했다[4]. 일반적으로 부정맥 감지를 평가하는 데 사용할 수 있는 데이터로, 심전도와 관련있는 딥러닝 분야에서 많이 사용되고 있다. 이 데이터에는 피험자 47명으로부터 수집한 2채널 구성의 24시간 ECG 기록을 30분동안 발췌하여 총 48개로 구성되어 있으며, 본 논문의 연구에서 기존 BeatGAN의 연구에 이용하던 데이터를 사용하였다. 총 데이터에는 97,568개의 비트와 2,860만의 시간 틱이 포함되어 있다.

4.3 연구 결과

크게 세 가지 방법으로 나누어서 연구를 진행했다.

- Feature distillation loss를 총 손실에 통합하는 방법
  - feature distillation loss를 총 손실에 통합시키고 train을 시킨 후 eval한 결과이다. auc는 약 0.9119로 사전에 훈련된 teacher model보다 다소 좋지 않은 성능을 보여주었다.

```
##############################
########  Result  ###########
ap:0.9018369128817183
auc:0.9340727981965814
best th:0.009869868867099285 --> best f1:0.8056621880998082
```

- Feature distillation loss과 Knowledge distillation loss를 총 손실에 통합하는 방법
  - Knowledge distillation loss 대신 feature distillation loss를 총 손실에 통합시키고 train을 시킨 후 eval한 결과이다. auc는 약 0.9119로 좋지 않은 성능을 보여주었다.

```
##############################
########  Result  ###########
ap:0.8875291196398069
auc:0.9119630947632182
best th:0.004932912532240152 --> best f1:0.7937380379840016
```

- Knowledge distillation loss를 총 손실에 통합하는 방법
  - Knowledge distillation loss를 총 손실에 통합시키고 train을 시킨 후 eval한 결과이다. Auc는 약 0.9464로 가장 좋은 성능을 보여주었다.

```
##############################
########  Result  ###########
ap:0.9134274215853668
auc:0.9464238429047412
best th:0.005571348126977682 --> best f1:0.8223677964492547
```

표 1은 기존의 BeatGAN 모델로 사전에 훈련을 시킨 후에 테스트한 결과와, 사전 훈련된 교사 네트워크를 이용하여 Knowledge Distillation 기법으로 학생 네트워크를 훈련시킨 후 테스트한 결과를 표로 나타낸 것이다. 표 1에서 볼 수 있듯이 기존의 모델보다 자가 개선을 수행하는 SI-BeatGAN의 정확도가 향상되었음을 알 수 있다. 그림 12는 BeatGAN과 SI-BeatGAN의 ROC 비교 그래프이며 마찬가지로 우리가 제안한 모델의 정확도가 더 높은 결과를 보여주고 있다.

표 1. ECG 데이터에서 BeatGAN과
지식증류가 수행된 SI-BeatGAN의 성능 차이

방법	AUC	AP
BeatGAN	약 0.937242	약 0.904847
SI-BeatGAN	**약 0.946423**	**약 0.913427**

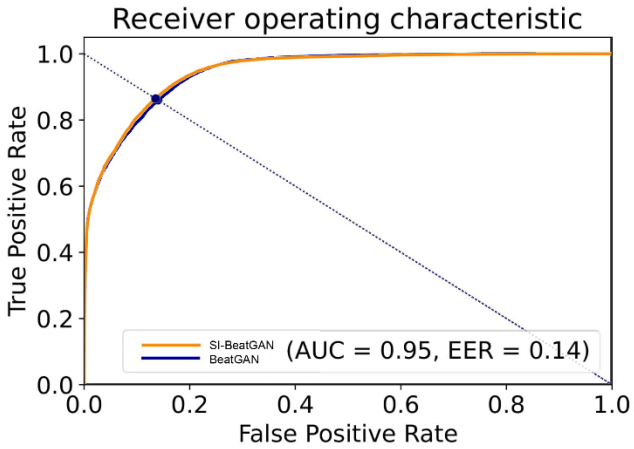


그림 12. ROC 비교 그래프

5. 결론

5.1. 기대효과

기존의 연구 및 논문을 통해 검증된 방식이자, 공개된 코드인 BeatGAN과 FRSKD를 접목시킴으로써, 좀 더 정확하고 많이 학습된 모델을 얻을 수 있다. 1차 목표는 FRSKD를 통해 다양한 지식을 학습하게 함으로써 BeatGAN의 경험 및 성능을 증진시키는데 있다. 본 연구를 통해 개선될 BeatGAN은 FRSKD를 활용하여 상위 모델로부터 증류되고 정제된 데이터를 학습하여 예상하기 어려운 비정상적인 심전도에서도 부정맥 여부를 좀 더 정확하게 판별할 수 있을 것이다. 즉, 본 연구를 통해 BeatGAN의 AUC를 높일 수 있으리라 기대할 수 있다.

5.2. 추후 연구 방향

본 프로젝트에서는 기존 BeatGAN 모델의 일부분만을 FRSKD의 지식 증류를 접목하는 방식으로 수정하여 개선하였고, 그 결과 기존의 BeatGAN 모델보다 상향된 AUC 값을 얻을 수 있었다. 그러나 연구 과정에서 제기되었던 개선 방안들을 모두 적용할 수는 없었고, 넓은 부분에서의 개선을 진행할 수는 없었다. 따라서, 향후에는 프로젝트의 모델에서 개선하지 못 했던 부분을 수정하여 문제점을 극복하고 효율을 높일 수 있고, 최근 연구가 활발해짐에 따라 새로이 등장하고 있는 다른 지식 증류 모델들을 연구하고 본 프로젝트에 추가적으로 도입해 봄으로써 더 나은 방향의 개선을 추구할 수 있을 것이다.

딥러닝 알고리즘은 최근 사람들의 관심사가 늘어남에 따라 다양한 연구와 개발이 진행되고 있고, 그렇기에 앞으로도 개발이 활발할 것이라고 기대할 수 있다. 따라서 본 프로젝트 또한 다양한 연구들을 접목하고 도입해 봄으로써 추가적으로 모델의 AUC를 높일 수 있는 가능성이 열려 있다고 볼 수 있다.

이러한 방식으로 프로젝트 모델의 AUC를 더 높이고 계속적으로 개선해 나간다면, 실제 의료 환경에서 본 프로젝트의 모델을 통해 환자의 ECG 데이터를 수집하고 분석함으로써, 환자의 이상 심전도를 더욱 빠르고 정확하게 감지해낼 수 있을 것이라고 기대할 수 있다.

6. 참고문헌

[1] Zhou B, Liu S, Hooi B, Cheng X, Ye J. BeatGAN: Anomalous Rhythm Detection using Adversarially Generated Time Series, Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence Main track. Pages 4433-4439, 2019

[2] Ji M, Shin S, Hwang S, Park G, Moon I. Refine Myself by Teaching Myself : Feature Refinement via Self-Knowledge Distillation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10664-10673, 2021

[3] Thomas Schlegl, Philipp Seeböck, Sebastian M. Waldstein, Ursula Schmidt-Erfurth, Georg Langs. Unsupervised anomaly detection with generative adversarial networks to guide marker discovery. In International Conference on Information Processing in Medical Imaging, pages 146–157. Springer, 2017

[4] Samet Akcay, Amir Atapour-Abarghouei, Toby P.Breckon. Ganomaly: Semi-Supervised Anomaly Detection via Adversarial Training. arXiv preprint arXiv:1805.06725, 2018

[5] Alec Radford, Luke Metz, Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434, 2015

[6] George B Moody and Roger G Mark. The impact of the mit-bih arrhythmia database. IEEE Engineering in Medicine and Biology Magazine, 20(3):45–50, 2001