설명 가능한 딥 러닝을 이용한 심전도 데이터 분석 연구

유재상°, 김성태

경희대학교 컴퓨터공학과

jsyoo1996@khu.ac.kr st.kim@khu.ac.kr

Electrocardiogram data analysis study using explainable deep learning

Jaesang Yoo°, Seongtae Kim

Department of Computer Science and Engineering, Kyung Hee University

요 약

최근 AI 기술이 4 차 산업혁명의 핵심기술로 자리잡으면서, 관련 연구가 활발하게 진행되고 있다. 특히 기계학습분야에 많은 발전이 있었고, 다양한 모델들이 다양한 분야에서 실험적으로, 또 실용적으로 사용되고 있다. 의료분야에서도 쓰이고 있는데 심전도 데이터(ECG) 분석의 경우가 대표적이다. 기존에는 숙달된 인력이 여러 파형을 보고 분석해 결과를 추론해야 했지만, 딥 러닝 모델을 통해 이 과정을 단순화, 가속화할 수 있음과 동시에 높은 정확성을 갖게 한다. 하지만 동시에 한계점을 가지는데, 바로 기계학습 모델의 블랙박스 특성으로 인해 어떤 과정을 통해 결론을 도출하였는지 알기 어렵다는 것이다. 때문에 이에 관한 연구를 통해 결과의 신뢰성을 높이고, 판별을 위해 기존에 사용되지 않았던 중요한 특징을 역으로 발견할 수도 있다.

1. 서론

1.1. 연구배경

딥 러닝 모델의 높은 정확도와 가능성으로 인해 쓰이지 않는 곳을 찾아보기가 더 어려워졌다. 이는 기존에 인간이 많은 시간과 정성을 들여서 해야 했던 일을 쉽게 해내기 시작했고, 그를 넘어서 인간이 해내지 못했던 분야까지 해내고 있다. 하지만 모델이 점점 더 복잡해짐에 따라 블랙박스라는 특성을 갖게 되었는데, 결과의 정확도는 높지만, 어떻게 그러한 결과를 도출했는지는 들여다보기 어렵게 된 것이다.

이 맹점은 실패했을 경우의 리스크가 큰 분야일수록 크게 다가온다. 특히나 의료분야에서 활발히 연구되고 있는 ECG 분석의 경우가 그렇다. 심전도 데이터로 크게는 부정맥이나 빈혈의 가능성을 진단할 수 있는데, 실패할 경우 인명에 영향을 미친다. 또한, 환자의 경우에는 증상에 대해 설명 받을

권리가 있고, 의사의 경우 설명해주어야 할 의무가 존재하는데, 블랙박스 형태의 모델의 경우 여기서 벗어나 있다.

따라서 단순히 모델의 높은 정확도만을 목표로 할 것이 아니라, 동시에 특이점을 포착해서 실질적인 사용자(모델의 개발자가 아닌)가 이해하기 쉬운 정보로 표현할 수 있어야 한다. 이 과정이 잘 정착되면, 사용의 용이성뿐만 아니라, 딥 러닝 모델의 신뢰성을 확보할 수 있는 장치로 자리매김할 것이다.

1.2. 연구목표

Dense Autoencoder, Convolution Autoencoder, K-NN의 모델을 이용하여 진행한다. MITBIH arrhythmia database에서 정상 심박 데이터를 이용해 각 모델에 학습시키고, reconstruction 시에 정상 심박과 이상 심박의 error를 확인한다. zero-padding 된 data가 학습에 방해가 될 수 있으므로, interpolation 한 data를 병행 사용한다.

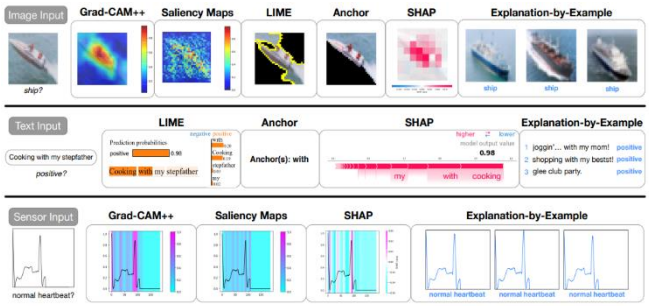
이를 기반으로 한 이상 탐지의 결과로, 데이터셋에서 가장 비슷한 심박을 추출해 증상 분류 모델의 효용성을 확인한다.

2. 관련연구

2.1. 딥러닝 모델의 설명 방법

서론에서 언급했듯 딥러닝 모델이 시간이 지날수록 높은 정확도를 가지게 되었으나, 동시에 모델이 어떤 특징을 포착해 판단을 냈는지 그 신뢰성은 떨어지게 되었다. 모델의 구조가 복잡해 사용자가 직관적으로 알 수 없게 된 것이다. 때문에 자연스럽게 딥러닝 모델의 설명 방법에 관한 연구도 이루어지게 되었다.

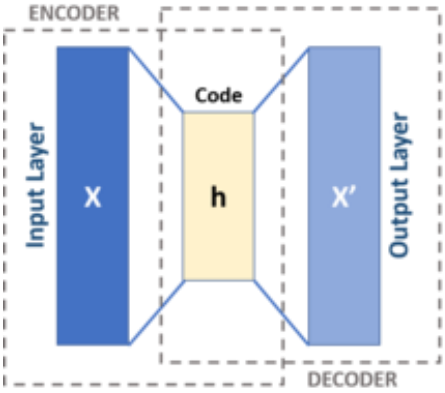
간단한 방법으로는 훈련 데이터 중 가장 인접한 데이터 값을 표현하는 방법부터, 훈련 데이터의 분포 중첩도를 그래프 형태로 출력해주는 방법 등, 다양한 데이터들에 따라 적합한 설명 모델들이 연구되고 있다.



[그림 1] 다양한 데이터 타입과 그에 따른 설명 방법 **[2]**

2.2.1. Autoencoder

Autoencoder는 비지도 학습 모델로, 입력 받은 데이터를 학습해 입력된 데이터에 최대한 가까운 데이터를 생성하는 것이 목표이다. 쉽게 말하면 입력을 예측하는 모델로, 주로 이상 감지, 혹은 노이즈 제거에 사용된다. 이상 감지의 경우 예측한 입력 값과 실제 입력 값의 비교를 통해 이상을 감지하는 방식으로 사용된다.



[그림 2] 기본적인 형태의 Autoencoder 구조

Autoencoder의 기본적인 형태는 [그림2]와 같이 입력 받은 데이터를 압축시키는 Encoder와 압축된 데이터를 복원하는 Decoder의 구조로 이루어져 있다. 여기에서 파생된 Convolution Autoencoder, LSTM Autoencoder 등의 모델이 존재한다.

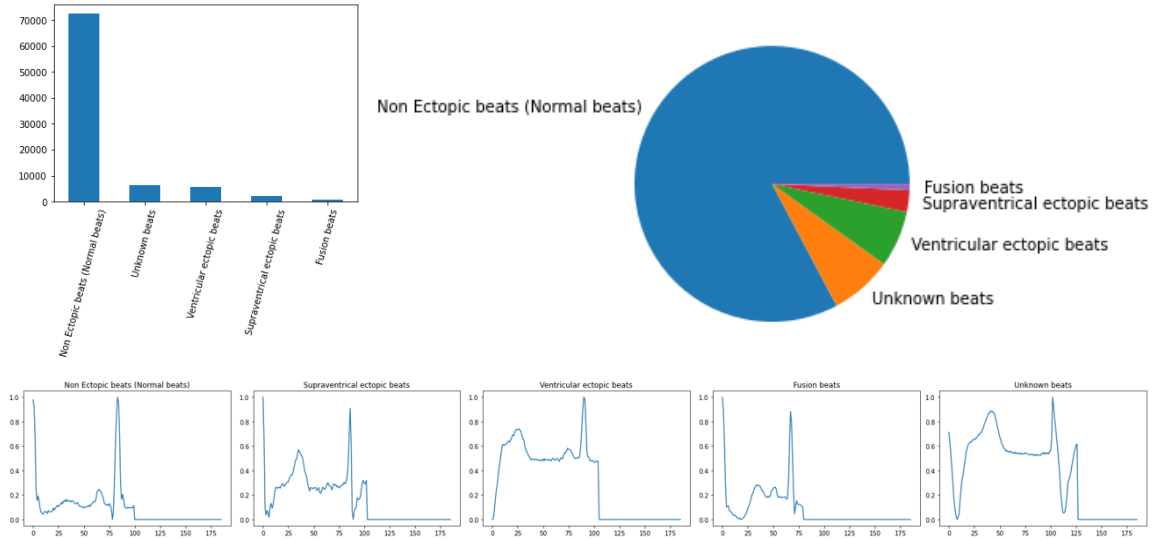
2.2.2. K-NN

K-최근접 이웃(K-Nearest Neighbor, KNN)은 지도 학습 알고리즘 중 하나이다. 어떤 데이터가 주어지면 그 주변의 데이터를 살펴본 뒤 더 많은 데이터가 포함되어 있는 범주로 분류하는 방식이다. K-NN의 특징은 훈련이 따로 필요 없다는 것이다. 훈련 데이터를 기반으로 모델을 만들고 테스트 데이터로 테스트를 하는 방식이다.

3. 프로젝트 내용

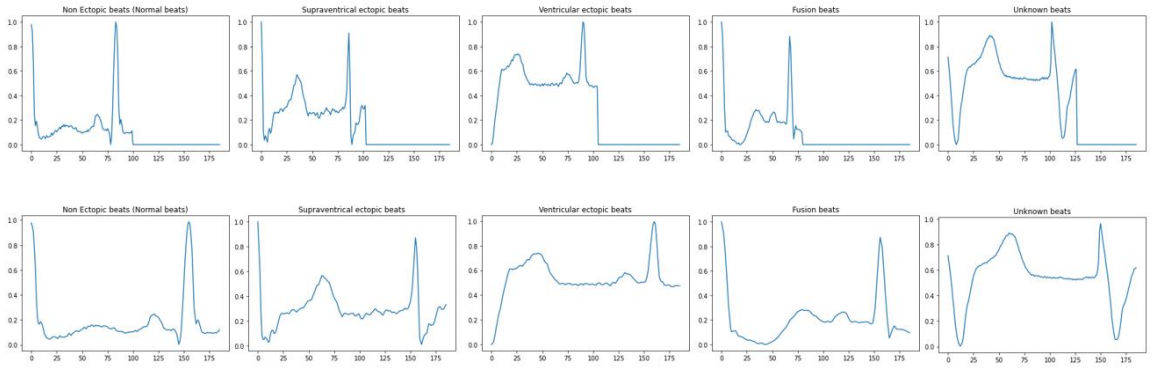
3.1. ECG dataset 분석 및 Interpolation

가장 널리 쓰이는 심전도 데이터는 MITBIH arrhythmia database이다. 정상 심박을 포함한 5가지 케이스로 나뉘어 있으며 그 크기는 다음과 같다.



[그림 3] 데이터의 개요와 각 케이스의 파형

Autoencoder는 정상 비정상만을 결정하기 위한 모델로, 1~4로 라벨링 되어있는 비정상 케이스들을 1로 통일 시켜 bool의 형태로 사용하였다. 추가적으로, 각 row의 길이가 일정하지 않고, 남는 data가 zero padding 되어 있기 때문에, 학습에 영향을 미칠 수 있기에 최대 길이인 187로 interpolation한 데이터를 병행 사용했다.



[그림 4] 원본 데이터와 interpolation 한 데이터의 비교

3.2. 이상 탐지 모델

3.2.1. Dense Autoencoder

수집한 데이터 중 정상 데이터만을 사용하여 학습한다. 한 개의 row는 187틱의 심박 정보와 1개의 라벨로 되어있는데, 결과적으로 이 심박 정보를 넣을 경우 정상 비정상 라벨을 결정해주는 모델을 설계한다.

```
self.encoder = tf.keras.Sequential([
  Dense(32, activation="relu"),
  Dense(16, activation="relu"),
  Dense(8, activation="relu")])

self.decoder = tf.keras.Sequential([
  Dense(16, activation="relu"),
  Dense(32, activation="relu"),
  Dense(187, activation="sigmoid")])
```

[그림 5] Dense Autoencoder의 구조

기본적으로 Autoencoder는 이와 같이 encoder와 decoder로 이루어져 있고, encoder에서는 data의 압축을, decoder에서는 복원을 한다. 정상 데이터만으로 학습을 하게 되면, autoencoder가 정상적인 데이터의 복원 분포를 익히게 되어, 비정상 데이터를 입력 받았을 때, 정상 데이터처럼 복원하게 되고, 이때 발생하는 오차로 이상 탐지를 하게 된다.

Autoencoder의 출력값은 정상 비정상 판단이 아닌, 복원된 심박 데이터이므로, 추가적으로 threshold를 지정해, 어느 수준의 error부터 비정상으로 판단할 것인지 기준을 세워주어야 한다. 여기서는 원본데이터와 복원데이터의 손실의 차이 분포를 확인하고, 평균손실 + 손실의 표준편차로 threshold를 지정했다. 높은 정확도를 위해서는 미세한 조정이 추가적으로 필요하며, 이를 위해 threshold에 따른 ROC score도 함께 확인한다.

3.2.2. Convolution Autoencoder

```
self.encoder = tf.keras.Sequential([
  Input(shape=(187, 1)),
  Conv1D(filters=64, kernel_size=7, padding='same', strides=2, activation="relu"),
  Conv1D(filters=32, kernel_size=7, padding='same', strides=2, activation="relu"),
  Conv1D(filters=16, kernel_size=7, padding='same', strides=2, activation="relu")])

self.decoder = tf.keras.Sequential([
  Conv1DTranspose(filters=16, kernel_size=7, padding='same', strides=2, activation="relu"),
  Conv1DTranspose(filters=32, kernel_size=7, padding='same', strides=2, activation="relu"),
  Conv1DTranspose(filters=64, kernel_size=7, padding='same', strides=2, activation="relu"),
  Conv1D(filters=64, kernel_size=6, strides=1, activation="relu"),
  Conv1DTranspose(filters=1, kernel_size=7, padding='same')])
```

[그림 6] Convolution Autoencoder의 구조

앞선 Dense Autoencoder에서 Dense Layer를 겹겹이 쌓아 모델을 만들었고, 여기서는 1차원 벡터 데이터의 Convolution에 쓰이는 Conv1D Layer로 대체되었다. encoder에서 홀수의 input shape을 주어서 decoder의 결과값과 오차가 생기게 되었는데, 이 부분을 보완하기위에 마지막에 커널 사이즈 6의 Conv1D Layer를 추가해 맞춰주었다.

3.3. 증상 분류 모델

앞의 Autoencoder에서 비정상 판별을 했다면, 이 부분에서는 KNN을 이용해, 갖고있는 데이터 내에서 가장 거리가 가까운 심박을 보여주고, 입력 받은 데이터의 증상 분류를 돕는다. 몇 개의 훈련 표본을 확인할 것인지 정하는 상수 K 값을 지정하고, 일반적인 KNN처럼 가장 가까운 K개의 데이터의 증상을 확인해 어떤 증상인지를 판별해주는 모델이 아니고, 해당 데이터들을 바로 표현해 사용자가 눈으로 확인할 수 있도록 한다.

```
knn_ecg = knn_model(3, anomal_party, y_temp)

I = knn_ecg.kneighbors([test_data[for_count]], n_neighbors=3, return_distance=False)
knn_pred = knn_ecg.predict([test_data[for_count]])
```

[그림 7] KNN 모델

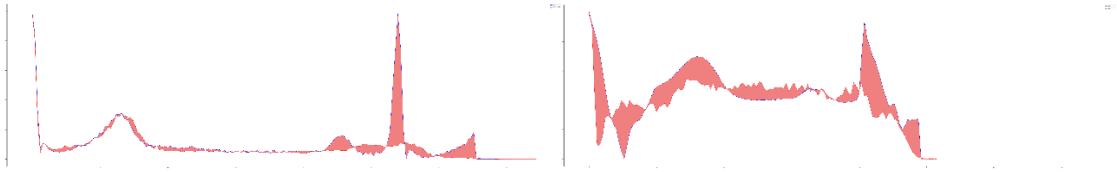
데이터의 양이 충분하지 않고, 불균형한 점도 있고, 사용자에게 비슷한 심박 데이터를 보여주는 것이 주 목적이기에 K의 크기를 크게 잡지 않고 3으로 설정하고 진행했다.

4. 프로젝트 결과

4.1. 연구 결과

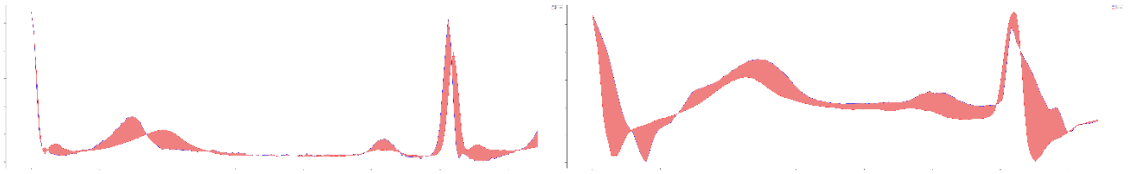
4.1.1. 이상 탐지 모델

먼저 Dense autoencoder를 사용했을 경우의 error는 아래와 같았다.



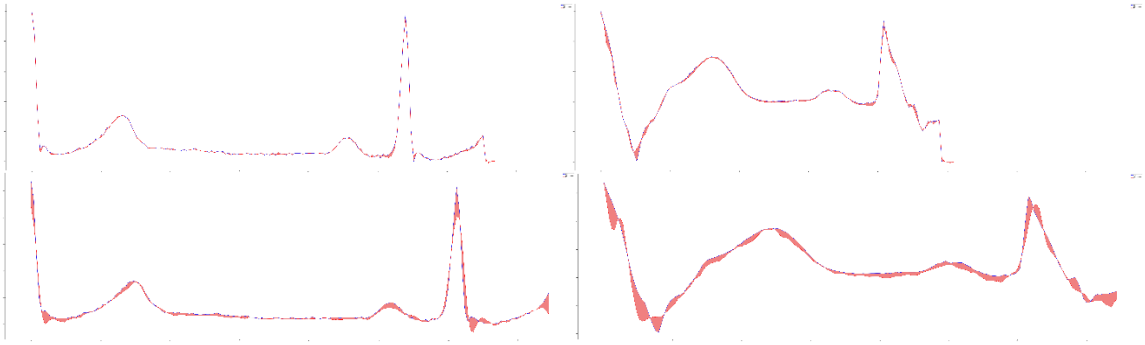
[그림 8] Dense autoencoder를 이용한 정상 심박(왼)과 비정상 심박(오)의 reconstruction 결과

각 이미지에서 파란 선은 원본 데이터, 붉은 선은 복원된 데이터이다. Dense를 활용하기 때문에, 짧은 구간에 극적으로 데이터의 고저가 바뀐 경우를 잘 캐치하지 못하는 경향이 보인다. 좌측이 정상 심박이고 우측이 비정상 심박인데, 확연하게 우측에서 error가 두드러지는 모습을 볼 수 있다. 추가적으로 interpolation한 경우를 같이 보면 아래와 같은데, 좀 더 원본의 데이터에 가깝게 복원되는 것을 볼 수 있다. 특히, 비정상 심박일 경우, error가 두드러지는 구간을 특정할 수 있고, 이를 근거로 사용자가 추가적인 분석을 할 수 있다.



[그림 9] Dense autoencoder에 interpolation된 데이터를 사용한 경우의 reconstruction 결과

이어서 Convolution autoencoder를 사용했을 경우에는 단순한 Dense보다 더 높은 복원률로 인해 error가 확연히 줄어든 것을 확인할 수 있었다. 동시에 원본 데이터와 interpolation 데이터를 사용한 경우간에 눈에 두드러지는 차이가 보이지 않았다.

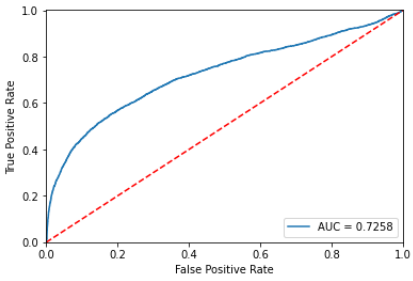


[그림 10] Convolution autoencoder 정상 심박(왼)과 비정상 심박(오)의 reconstruction 결과

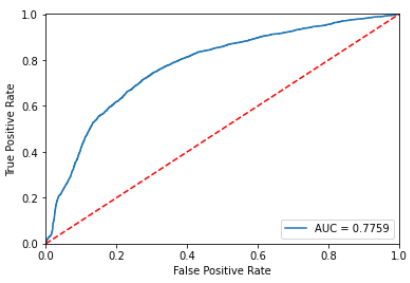
한 개의 예시로만 모델의 성능을 판별할 수는 없기 때문에, 추가적인 성능 검증을 위해 AUC SCORE를 비교했다. AUC – ROC Curve는 수신자 조작 특성(Receiver Operating Characteristic)으로 X축은 false positive rate이고, Y축은 true positive rate를 나타낸다. 이 경우, 각각 정상심박일 때 정상이라고 예측했을 경우, 비정상 심박일 때 비정상이라고 예측했을 경우이다. ROC Curve가 왼쪽 위에 가까워 AUC(Area Under Curve)가 1에 가까워질수록 오류가 적은 좋은 모델이라고 할 수 있다.

Pure Data

Dense Autoencoder

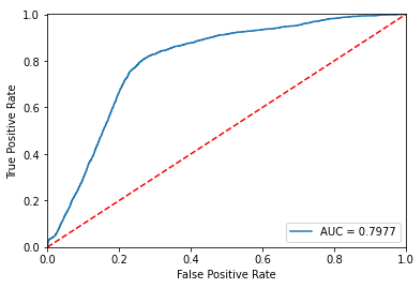


Convolution Autoencoder

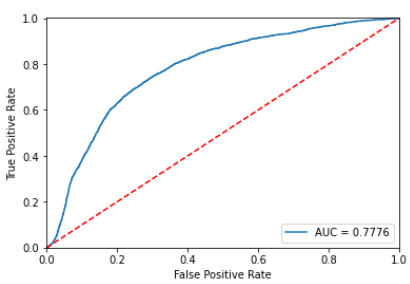


Interpolationed Data

Dense Autoencoder



Convolution Autoencoder

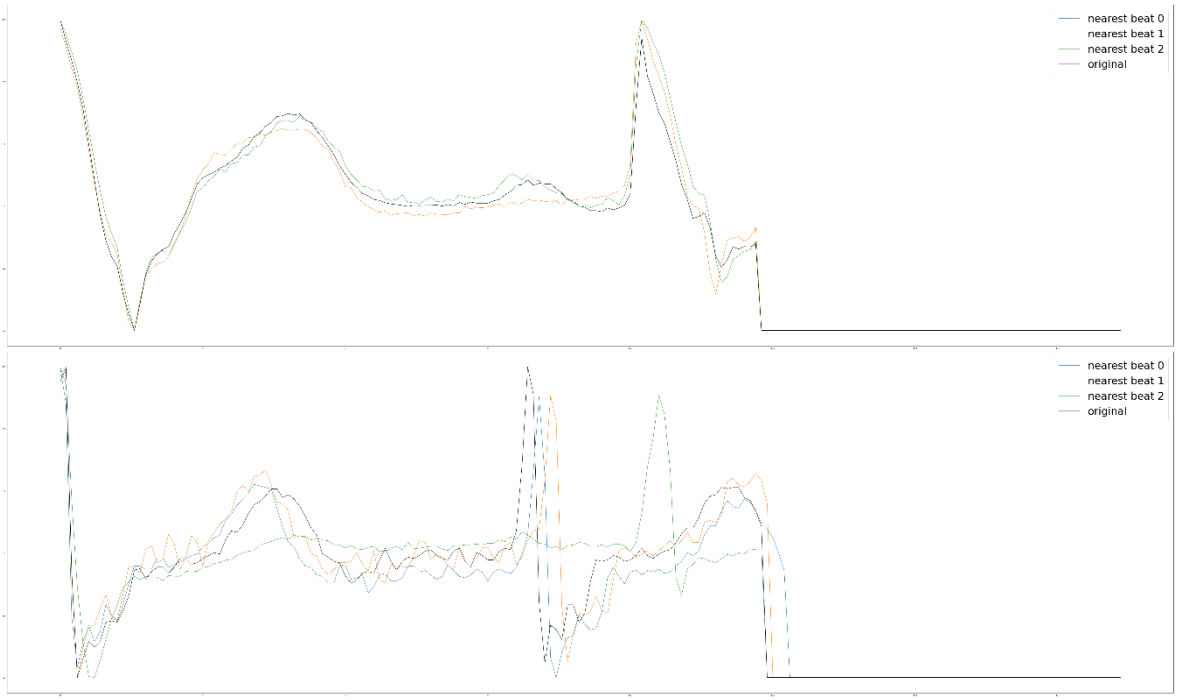


[그림 11] 각 모델에 대한 AOC – ROC Curve

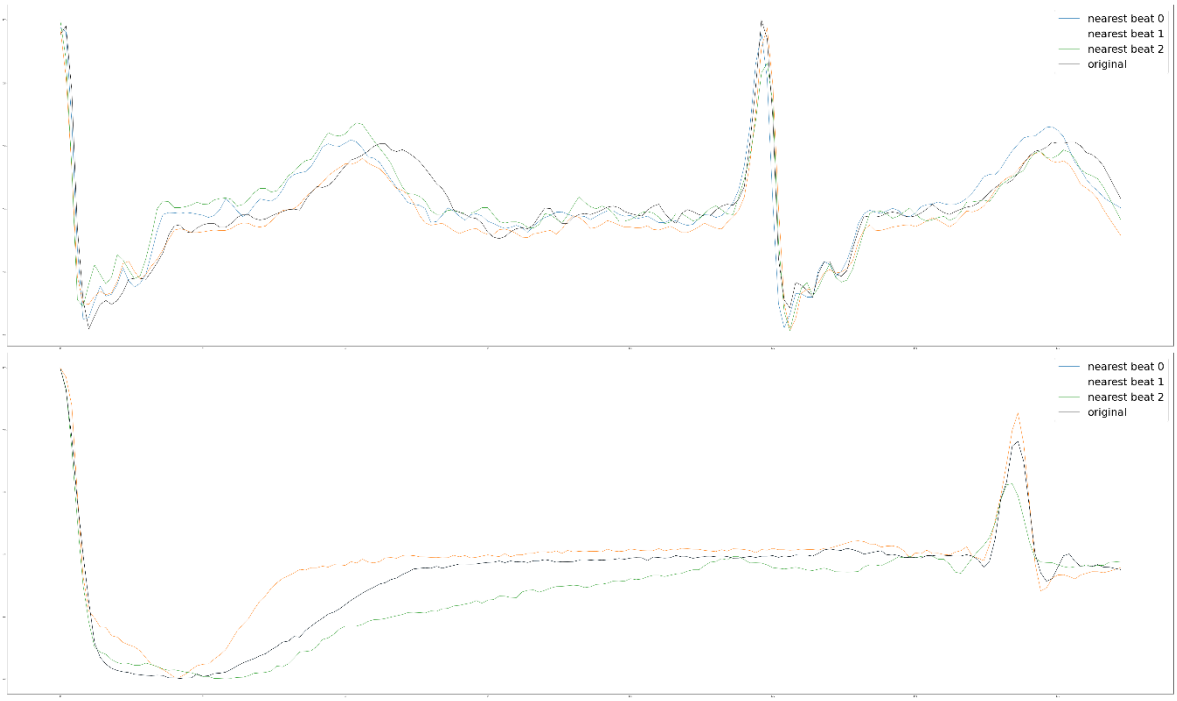
실제로 Dense Autoencoder의 경우 데이터의 Interpolation 여부가 상당한 영향을 끼쳤음을 확인할 수 있다. 반면 Convolution Autoencoder는 이에 영향을 크게 받지 않고 보편적으로 일정한 수준의 성능을 보임을 알 수 있다.

4.1.2. 증상 분류 모델

knn의 경우 Convolution Autoencoder의 결과값을 토대로 비정상이라고 판별했을 경우에 비정상 class 중 가장 비슷한(가까운) 파형의 심박을 찾아내도록 하였다.



[그림 12] 원본 데이터를 사용했을 경우의 KNN



[그림 13] Interpolation 데이터를 사용했을 경우의 KNN

KNN은 데이터 의존도가 높은 알고리즘인데, 이로 인해서, 전혀 다른 파형의 심박을 보여주는 경우가 있다. Interpolation 데이터를 사용했을 경우, 심박의 길이라는 feature에 자유도가 생겨 더 좋은 성능을 보였다. 결과를 봤을 때, 비정상 class 중에서도 가장 볼륨이 큰 Ventricular ectopic beats와 Unknown beats로 예측하는 경우가 많았다. 때문에 충분한 데이터가 갖춰지지 않았다면, 분류 모델로 사용하기는 어렵고, 보조 지표로 사용할 수 있을 것이다.

5. 결론

5.1. 기대효과

본 논문에서는 설명가능한 딥 러닝 모델을 이용해 심전도 데이터의 분석을 진행해, 개발자를 넘어서 실제 사용자가, 이 경우에는 의료인과 환자가 쉽게 건강의 이상유무를 판별하고, 딥 러닝 모델 신뢰성 저하의 특징인 블랙박스를 해소했다. 사용자는 autoencoder 모델의 결론 도출과정으로 눈으로 확인할 수 있기에 과정과 결과에 대한 이해가 쉬워지고, 추가적으로 knn으로 실제 이상 증세의 예시와 비교가 가능해 단순 이상 탐지를 넘어서 증상 진단과 그 과정에서의 오류를 사용자가 직접 검증할 수 있다. 이를 기반으로, 의료시설이 미흡한 곳에서도 심전도의 측정과 분석이 용이해지고, 보편적 인간의 건강 증진으로 이어질 수 있다.

5.2. 추후 연구 방향

사용한 데이터를 포함해, 공개되어 있는 심전도 데이터가 많지 않고, 존재하는 이상 증상은 많은 반면에 실존하는 데이터는 몇 종류 제공하고 있지 않거나, 제공하고 있는 데이터도 불균형 하다. 통합적인 의료 데이터베이스 구축이 가능하다면, KNN으로 제공하는 증상의 분류 정확도도 올릴 수 있으며, autoencoder에도 비정상 분포를 학습시키는 등의 새로운 시도를 해볼 수 있을 것이다.

추가적으로, Autoencoder에서 error가 큰 구간의 localization을 통해 중요한 특이점의 값에 가중치를 주어 knn에 입력시켰을 경우의 성능 개선여부도 확인해 볼 수 있다.

여기까지는 개발자가 단순하게 존재하는 데이터로 만들어낸 것이고, 실질적으로 의료인이 심전도 데이터를 판단하는 기준인 심박에서의 PQRST feature와 심박 길이와 같은 새로운 feature로 학습을 한다면 더 뛰어난 성능의 모델을 만들어 낼 수 있을 것이다.

참고문헌

[1] Zhou, B., Liu, S., Hooi, B., Cheng, X. and Ye, J., 2019, August. BeatGAN: Anomalous Rhythm Detection using Adversarially Generated Time Series. In *IJCAI* (pp. 4433-4439)

[2] Jeyakumar, J.V., Noor, J., Cheng, Y.H., Garcia, L. and Srivastava, M., 2020. How can i explain this to you? an empirical study of deep neural network explanation methods.

[3] Kuznetsov, V.V., Moskalenko, V.A., Gribanov, D.V. and Zolotykh, N.Y., 2021. Interpretable Feature Generation in ECG Using a Variational Autoencoder. *Frontiers in Genetics*, *12*.

[4] Ribeiro, A.H., Ribeiro, M.H., Paixão, G.M.M. *et al.* Automatic diagnosis of the 12-lead ECG using a deep neural network. *Nat Commun* 11, 1760 (2020).

[5] Golany, T., Radinsky, K. and Freedman, D., 2020, November. Simgans: Simulator-based generative adversarial networks for ecg synthesis to improve deep ECG classification. In *International Conference on Machine Learning* (pp. 3597-3606). PMLR.