Classification of Symptomatic Patent Ductus Arteriosus with RNN & Data Augmentation

연세대학교 컴퓨터과학과 소프트웨어종합설계 최종 보고서

5분반 **휴먼러닝** 팀

[redacted]

2021-12-15

Introduction

Machine Learning(ML)을 의학적 진단에 사용하려는 시도는 심장 질환[1], 유방암[2] 등 다방면에서 이루어지고 있다. ML 진단 시스템은 의사가 보조적인 도구로 사용함으로써 오진을 줄이고 의료 비용을 감소시키도록 활용될 수 있다[3]. 그러나 의학적 진단에 ML을 사용할 때에 필요한 의료 데이터의 경우 적은 수의 샘플과 불균형한 라벨 비율, 낮은 데이터 품질 등의 문제에 부딪히기 쉽다. 특히나 최근 Speech Recognition, Image Classification 등에서 높은 성능을 보인 Deep Learning(DL)은 많은 샘플 수를 요구하여 희귀 질환의 경우 적용하기 어려웠다.

Patent Ductus Arteriosus(PDA)는 신생아에게 발병하는 희귀 심장 질환으로, 출생 이후에 동맥관이 닫히지 않아서 발생한다. 일반적인 경우에 동맥관은 출생 이후 곧바로 자연스럽게 닫힌다. 반면에 계속해서 열려 있게 되면 산소 공급에 지장이 생겨서 다양한 합병증이 발생할 수 있다. 따라서, 동맥관이 닫히지 않은 신생아에 대해 의료진이 자연적으로 닫힐 것인지, 질환으로 이어질 것인지를 빠르게 판단하여 수술 등 조치를 취하는 것이 합병증을 예방하기 위해 중요하다.

연세대학교 디펜더블 컴퓨팅 연구실에서는 다양한 ML 모델을 활용하여 PDA의 Early Prediction을 수행할 때 의료진을 보조할 수 있는 시스템을 개발하였다. 해당 연구는 Naive Bayes, Linear Regression, Random Forest(RF), kNN 4가지 모델을 사용하였으며, 일자별 Lab Test, Clinical Record

데이터셋으로 학습시켰다. 결과적으로 Decision Accuracy를 최대 27.2%p 향상시키고 Decision Making을 최대 2.1일 가량 앞당기는 등의 성과를 보였다. 그러나, 가장 높은 성능을 보인 Random Forest 모델의 경우 시계열 데이터를 충분히 활용하지 못한다는 한계가 존재했다.

본 연구는 같은 데이터셋을 사용했을 때 DL 모델이 성능을 향상시킬 수 있는지를 알아볼 것이다. 첫째로, 데이터셋이 시계열의 특성을 가진 것을 고려하여 Recurrent Neural Network(RNN)에서 비롯한 모델을 사용한다. 둘째로, RNN을 학습하기에 상대적으로 적은 샘플 수(n=409)를 보완하기 위해 Data Augmentation, Feature Selection, Ensemble을 적용하여 성능을 더 높일 수 있는지 알아본다.

Related Work

Data Augmentation

(1) Random transformation-based data augmentation

이미지 데이터에 Cropping, Flipping, Noise Addition 등의 방법으로 랜덤하게 데이터를 생성해내는 것과 마찬가지로 Time Series에도 Data Augmentation을 적용할 수 있다. 그 중 한 방법은 Random Transformation을 통해서 원본 Time Series 데이터의 일부가 조작된 데이터를 생성하는 방식이다. 이러한 Random Transformation에는 Magnitude Domain Transformation, Time Domain Transformation, Frequency Domain Transformation 세 종류가 있다. Jittering과 Scaling은 다양한 형태의 Time Series 데이터에 간단한 구현으로 적용 가능한 Data Augmentation 방법으로, Magnitude Domain Transformation에 속한다[4].

Jittering은 원본 데이터에 노이즈를 더해서 새로운 데이터를 만들어내는 방식이다.

$$\mathbf{x}' = x_1 + \epsilon_1, \ldots, x_t + \epsilon_t, \ldots, x_T + \epsilon_T,$$

$$\varepsilon_i \sim N(0, \boldsymbol{\sigma}^2)$$

입력 데이터 X의 각 Time Step T에 대해서 서로 다른 노이즈 ε_i를 더한다. 이 때 ε_i는 표준편차가 $\boldsymbol{\sigma}$인 Gaussian Noise이다. $\boldsymbol{\sigma}$는 미리 정해진 Hyperparameter로 Data Augmentation의 성능에 영향을 준다. Jittering을 통해 노이즈를 더하는 방법은 Neural Network의 일반화를 증대할 수 있는 방법 중 하나이다.

반면에 Scaling은 원본 Time Series에 일정한 비율을 곱하여 새로운 데이터를 만들어낸다.

$$\mathbf{x}' = \alpha x_1, \ldots, \alpha x_t, \ldots, \alpha x_T,$$

$$\boldsymbol{\alpha} \sim N(1, \boldsymbol{\sigma}^2)$$

α는 평균이 1이고 표준편차가 **σ**인 Gaussian Noise이다.

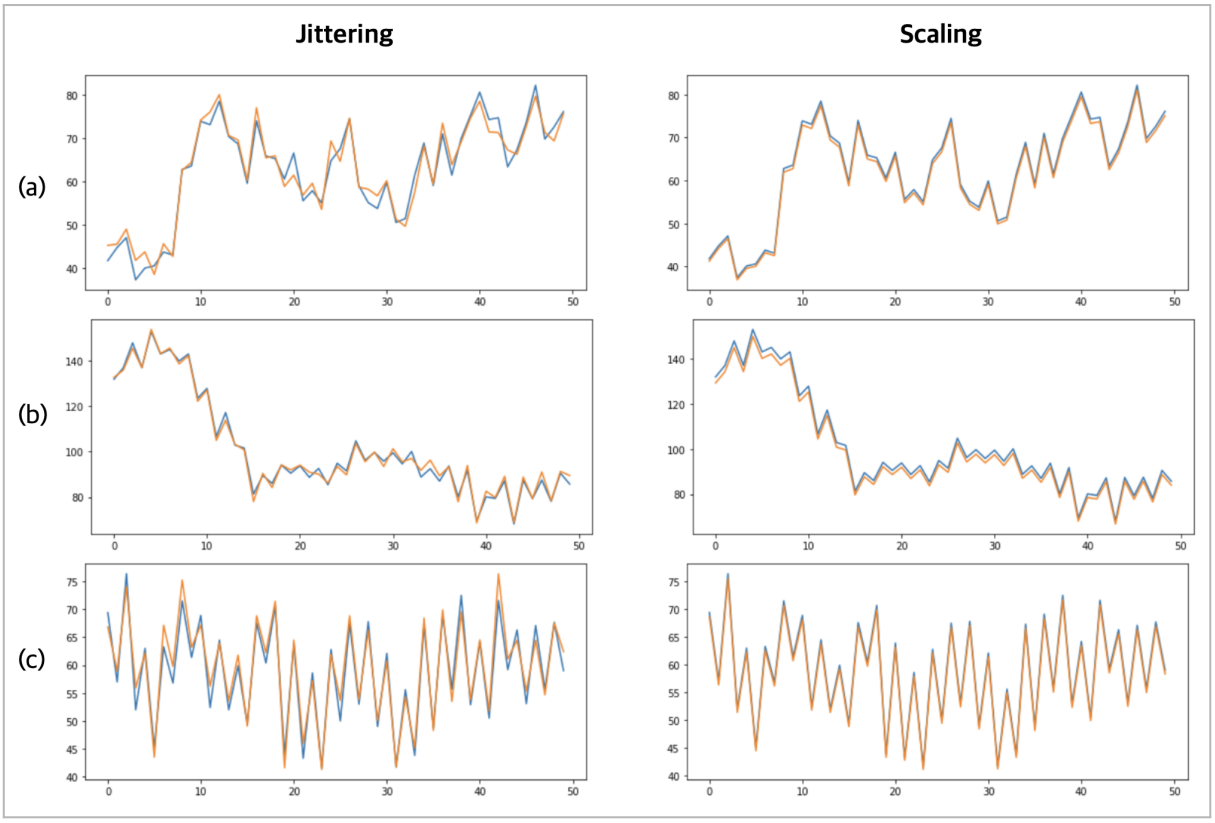

Figure 1. Magnitude Domain Transformation을 시각화한 그래프. 세 개의 Time Series에 대해 각각 Jittering과 Scaling을 적용했다. 파란색은 원본 Time Series이며 주황색은 Augment된 결과이다.

Reference [4]에서 Neural Network 학습에 대해 여러 Time Series Data Augmentation을 비교한 결과에 의하면 Jittering과 Scaling은 다양한 모델과 데이터셋에 대해서 전반적으로 높은 성능을 보였다.

(2) Neural network-based generative models

최근 Neural Network을 사용한 다양한 Generative Model이 제시되었다. 그 중 Encoder-Decoder Network, Generative Adversarial Network 등 모델은 Time Series Data Augmentation에 사용될 수 있다. Generative Adversarial Network(GAN)는 Generator와 Discriminator 두 가지의 Neural Network의 Adversarial Training을 통해 함께 학습시키는 방법이다. Generator는 Training Data로부터 학습하여 랜덤한 z-vector를 샘플링 한 후 그 결과를 바탕으로 새로운 샘플을 만들어낸다. Discriminator는 Generator로부터 만들어진 샘플과 원본 샘플을 구별하는 역할을 한다. 이러한 원리를 통해 GAN은 Data Generation에 효과적으로 사용될 수 있고, 또한 Data Augmentation을 위해 사용되기도 한다[4].

medGAN은 프라이버시 침해 우려로 인해 Electronic Health Record(EHR) 데이터를 자유롭게 활용하지 못하는 점을 개선하기 위해서 고안된 GAN 모델이다. medGAN은 Synthetic EHR 데이터를 생성하여 실제 EHR 데이터에 접근할 수 없는 경우에도 예측 모델을 충분한 성능으로 학습시키는 것을 목표로 한다. medGAN의 구조는 Random Prior를 Feed-forward Network인 Generator에 넣어 Discriminator가 Fake Sample과 Real Sample을 구분하도록 학습한다는 점에서 기존 GAN[5]과 비슷하지만, 보다 중요도가 높은 Feature들을 찾고 이산적인 결과를 산출하도록 AutoEncoder를 활용하였다는 차이점이 있다.

$$\theta_d \leftarrow \theta_d + \alpha \nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \log D(\mathbf{x}_i, \bar{\mathbf{x}}) + \log(1 - D(\mathbf{x}_{\mathbf{z}_i}, \bar{\mathbf{x}}_{\mathbf{z}}))$$

$$\theta_{g,dec} \leftarrow \theta_{g,dec} + \alpha \nabla_{\theta_{g,dec}} \frac{1}{m} \sum_{i=1}^{m} \log D(\mathbf{x}_{\mathbf{z}_i}, \bar{\mathbf{x}}_{\mathbf{z}})$$

$$\text{where } \bar{\mathbf{x}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i, \quad \mathbf{x}_{\mathbf{z}_i} = Dec(G(\mathbf{z}_i)), \quad \bar{\mathbf{x}}_{\mathbf{z}} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_{\mathbf{z}_i}$$

추가적으로, GAN에서 Generator가 학습하는 과정에서 Real Sample의 분포를 외워 같은 결과물을 만들어내는 Mode Collapse 현상을 방지하기 위해 Discriminator에 위와 같이 Minibatch Sample들의 Average를 학습하도록 하는 Minibatch Averaging 기법을 사용하였다[6].

Feature Selection

ML에서 사용하는 데이터에서는 종종 학습에 크게 기여하지 않는 Feature가 포함되는 경우가 있다. 이런 경우에는 Feature Selection을 통해서 일부 Feature Set을 필터하여 학습 및 예측할 수 있다. Feature Selection은 각 Feature의 Importance를 측정하고 기준으로 삼아서 진행할 수 있다. 이러한 과정을 통해서 모델의 복잡성을 줄이고, 예측 성능을 높이는 것을 목표로 한다. Feature Importance의 지표로 사용될 수 있는 것 중 하나는 Random Forest(RF)에서 비롯한 Impurity-based Importance이다.

Reference [7]에서는 Convolution Neural Network를 학습시킬 때 사용하는 Feature를 선택하기 위해 RF로부터 Feature Importance를 추출하였고, 총 26개의 Feature 중 20개를 선별했다.

Ensemble

Ensemble은 작은 모델 여러 개를 종합하여 더 정확한 모델을 만들어내는 과정이다. Reference [8]의 연구에서 적은 데이터 수의 Fall Detection Dataset을 RNN으로 예측하기 위해서 Boosting, Stacking 등의 Ensemble을 적용하였고, 그 중 Stacking의 방식을 통해 성능 향상을 이루어냈다.

한편, Reference [9]의 연구에서는 심장병을 예측하기 위해서 다양한 종류의 모델을 Majority Voting Ensemble로 종합하였다. Majority Voting Ensemble은 각 모델이 예측값에 대해 한 표를 행사했을 때, 다수결로 나온 결과를 최종 예측값으로 채택하는 방식이다. 해당 연구에서는 Ensemble을 통해 각각 모델의 성능보다 높은 성능을 지닌 종합 모델을 제시했다.

Methodology

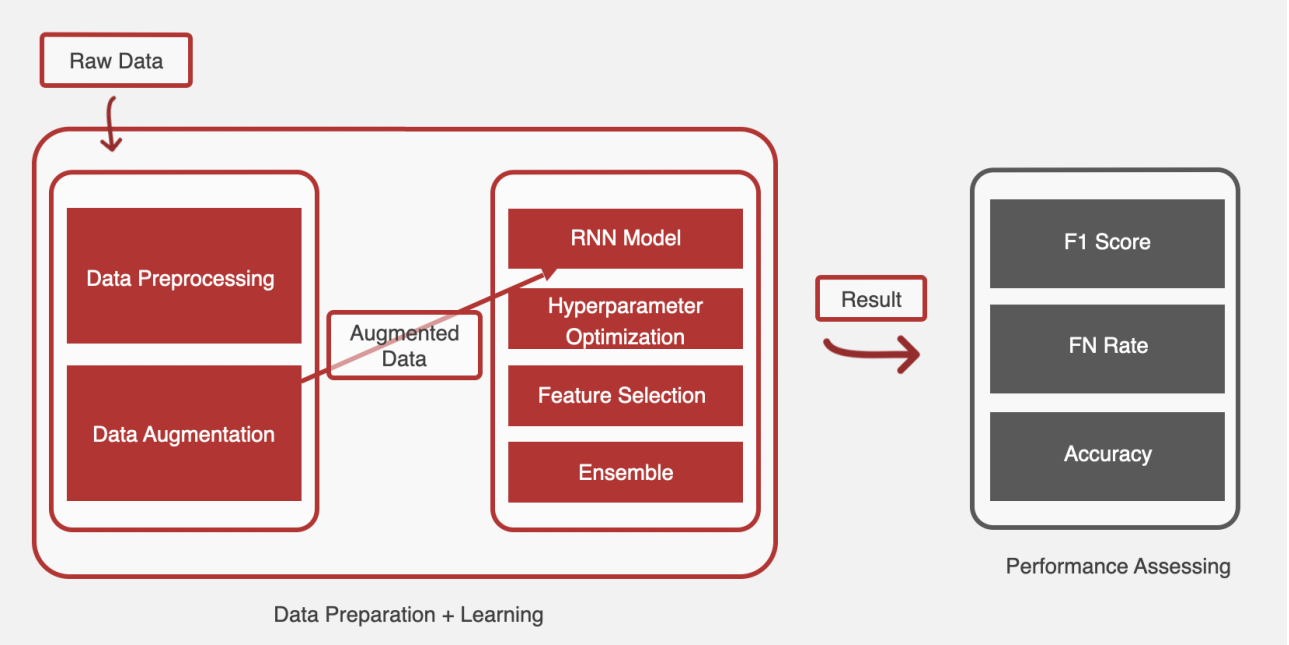

Figure 2. 연구의 전체 흐름을 정리한 Chart.

Dataset

학습을 위하여 총 406명의 환자로부터 얻어진 생체 정보 데이터를 활용하였다. 각 데이터는 19개의 Prenatal Factors, 80개의 Postnatal Factors를 포함하여 총 99개 Dimension이며, PR(Pulse Rate), BT(Body Temperature), RESP(Respiration) 등을 포함하였다. 환자 각각에 대해서 데이터의 형태는 Multivariate Variable Length Time Series이다. Time Series의 길이는 1~15일 사이에 분포하였다. 주목할만한 점으로, PDA로 진단된 환자는 1~15일의 길이를 가졌지만 PDA가 아닌 환자는 모두 15일의 길이를 가졌다.

Pre-processing은 다음과 같이 진행되었다. 첫째로, 각 환자에 대해서 N일자의 데이터를 총 N개로 분리하였다. 환자의 i번째 데이터셋은 1일자부터 i일자까지의 정보를 담은, 원본 Time Series의 부분 집합이다.

$$\{ [1, 2, 3] \} \rightarrow \{ [1], [1, 2], [1, 2, 3] \}$$

길이 3인 Time Series를 예시를 들면 위와 같이 데이터를 분리시킨 것이다. N일자 데이터로 분리시킨 목적은 학습시킨 모델이 Early Prediction을 수행할 수 있도록 하는 것이다. 즉, 의료진이 진단을 내려서 환자에 개입을 하기 전에 ML 모델이 예측하는 것을 목표로 한다. 둘째로, Variable Length Time Series를 학습에 용이하도록 Fixed Length로 고정하였다. Pre-zero Padding을 사용하여 모든 데이터의 길이를 15로 고정하였다. Pre-zero Padding은 주어진 데이터가 없는 구간을 모두 0으로 초기화하여 공간을 비워두는 작업으로, RNN 모델에 사용하기에 적합하다.

$$[1, 2, 3] \rightarrow [0, 0, \cdots, 0, 1, 2, 3]$$

길이 3인 Time Series에 Pre-zero Padding을 적용한 예시이다. 셋째로, Normalization을 진행했다. Min-Max Normalization을 통해 모든 데이터 값을 [1, 2] 사이에 위치하도록 조정하였다. 따라서 0으로 설정한 Zero-Padding과는 값이 구별된다.

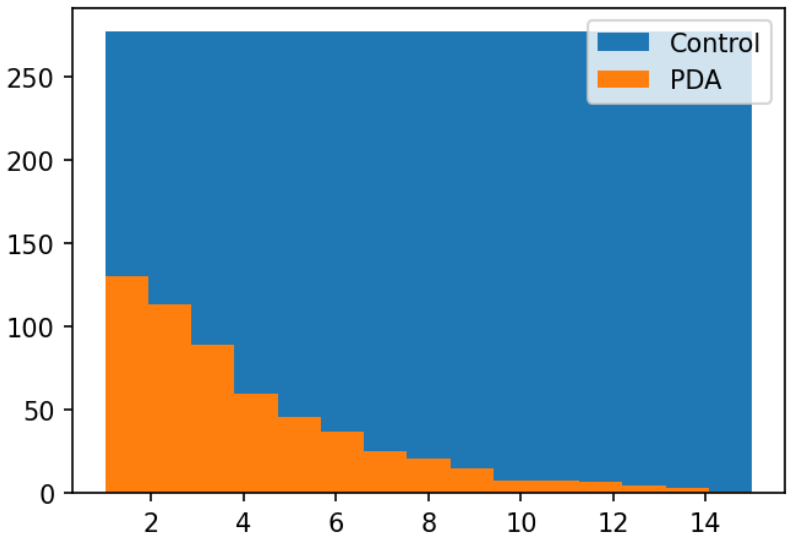

Figure 3. Pre-processing 이후 Control과 PDA 그룹의 Time Series 길이의 분포를 나타낸 히스토그램.

Model

RNN은 Hidden Layer들이 방향을 가진 Edge로 연결되어 Directed Cycle을 이루는 인공 신경망의 한 종류로, 순차적 데이터 처리에 적합한 알고리즘이다. 본 연구의 경우 EHR 데이터가 일자별로 주어지므로 RNN을 이용하여 Classification을 하는 것이 적합할 것이라고 판단하였다. 모델 설계를 위해 Pytorch에서 제공하는 `rnn` 모듈을 이용했다.

결과 파트에서는 아래의 4가지 RNN Variants의 성능을 비교한다.

(1) Vanilla RNN

가장 기본적인 Vanilla RNN 알고리즘이다. Input Layer의 벡터를 Hidden Layer에서 처리하여 Output Layer로 보낸 결과값을 활용하여 Classification을 진행한다.

(2) LSTM

LSTM[10]은 RNN의 단점을 보완한 신경망으로, Gradient Vanishing Problem을 방지하기 위해 Gradient가 그대로 전달되는 Cell State를 추가한다. 그리고 Forget, Input, Output을 조정하는 Gate를 만들어 각각의 Parameter을 학습하는 구조이다.

(3) GRU

GRU[11]는 Gated Recurrent Unit의 약자로, LSTM과 비슷하지만 Reset과 Update 두 개의 Gate를 가지고 있다.

(4) Attention-based RNN

RETAIN[12]은 **RE**verse **T**ime **A**ttent**I**o**N**의 약자로, 다른 Traditional ML 모델보다 정확도는 높지만 Interpretability가 낮은 범용 RNN Model의 단점을 보완하기 위해 고안된 모델이다. EHR Data를 타겟으로 하는 이 모델은 2개의 Attention을 사용하는데, 각 단계 시퀀스의 중요도를 나타내는 Visit-level Attention과 Data의 각 특성에 대한 중요도를 나타내는 Variable-level Attention이 있다. RETAIN에선 이 두가지 Attention과 함께 Time Series Data를 역으로 순회하는 RNN을 만든다. 이를 통해 질병의 발병 가능성과 그 해석을 함께 진행할 수 있는 모델이다.

Hyperparameter Optimization

딥러닝 모델에는 Hyperparameter와 Model Parameter가 있다. Training이 진행되는 동안 자동적으로 최적화되는 Model Parameter와 달리, Hyperparameter는 Model Training이 시작되기 전에 결정되어있어야 하며, 모델이 어떻게 학습할지에 영향을 미친다. Hyperparameter Optimization을 위한 다양한 라이브러리들이 통합되어 있고, 업계에서 주로 사용된다고 알려진 RayTune[13]을 활용하여 모델별로 최적의 Hyperparameter 조합을 찾아내고자 했다.

```
scheduler = ASHAScheduler(
    metric="loss",
    mode="min",
    max_t=max_num_epochs,
    grace_period=1,
    reduction_factor=2)
```

```
'batch_size': tune.choice([128, 256, 512, 1024]),
'lr': tune.loguniform(1e-5, 1e-2),
'dropout': tune.choice([0.2, 0.3, 0.4, 0.5]),
```

Figure 4. ASHAScheduler를 사용한 Hyperparameter 관련 예시 Code snippet.

Ray에서 Standard로 제안하는 ASHAScheduler를 사용하여 loss를 최소화하는 방향으로 학습 진행하였다. `reduction_factor`(반감기)는 2, `grace_period`는 100으로 설정하였다. `batch_size`는 128, 256, 512, 1024 / `learning_rate`는 0.01에서 0.00001 / `dropout_rate`는 0.2, 0.3, 0.4, 0.5의 옵션들에서 각각 랜덤 샘플링하여 4가지 모델에 대한 Hyperparameter Optimization을 진행하였다.

Feature Selection

Feature Selection을 위해서 Random Forest를 학습시키고, 모델에서 Impurity-based Importance를 추출하였다. Impurity는 RF가 분류할 때 사용하는 지표로, Feature의 불순도를 의미한다. 불순도가

낮다는 것은 어느 하나의 클래스에 Feature의 값이 특히 높거나 낮게 분포한다는 뜻이다. 즉, Impurity-based Feature Importance가 높으면 RF 모델이 해당 Feature를 사용하여 분류를 잘할 수 있다. 모든 Feature를 Feature Importance 순으로 정렬한 후, Top N개의 Feature만을 사용하여 학습 및 분류를 진행했다(N = 99, 95, 92, 88).

Ensemble

Random Initialization이 다른 총 5개의 모델을 학습시켜서 Majority Voting Ensemble을 적용하였다.

Performance Evaluation

Train과 Test의 데이터셋은 0.8:0.2로 분할하였다. 또한 Hyperparameter Optimization을 진행할 때 5-Fold Cross Validation을 적용했다.

각 모델의 성능은 F1 Score, Accuracy, FN Rate를 기준으로 비교하였다. F1 Score와 Accuracy는 높을수록 좋고, FN Rate는 낮을수록 좋다. 아래는 F1 Score, Accuracy, FN Rate를 구하는 식이다.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\ Score = 2\frac{Precision + Recall}{Precision \times Recall}$$

$$Accuracy = \frac{TP + TN}{P + N}$$

$$FN\ Rate = \frac{FN}{P}$$

Results

Performance Before Optimization

Model	RNN	LSTM	GRU	RETAIN	RF
F1 Score	0.7037	0.7428	0.7266	0.241	0.507

Figure 5. Hyperparameter Optimization을 진행하기 전 모델 별 F1 Score.

Hyperparameter Optimization

결과로 얻어진 최적 Hyperparameter는 다음과 같다. `batch_size`: 128, `dropout_rate`: 0.3, `learning_rate`: 0.0008131, `model`: LSTM.

Data Augmentation

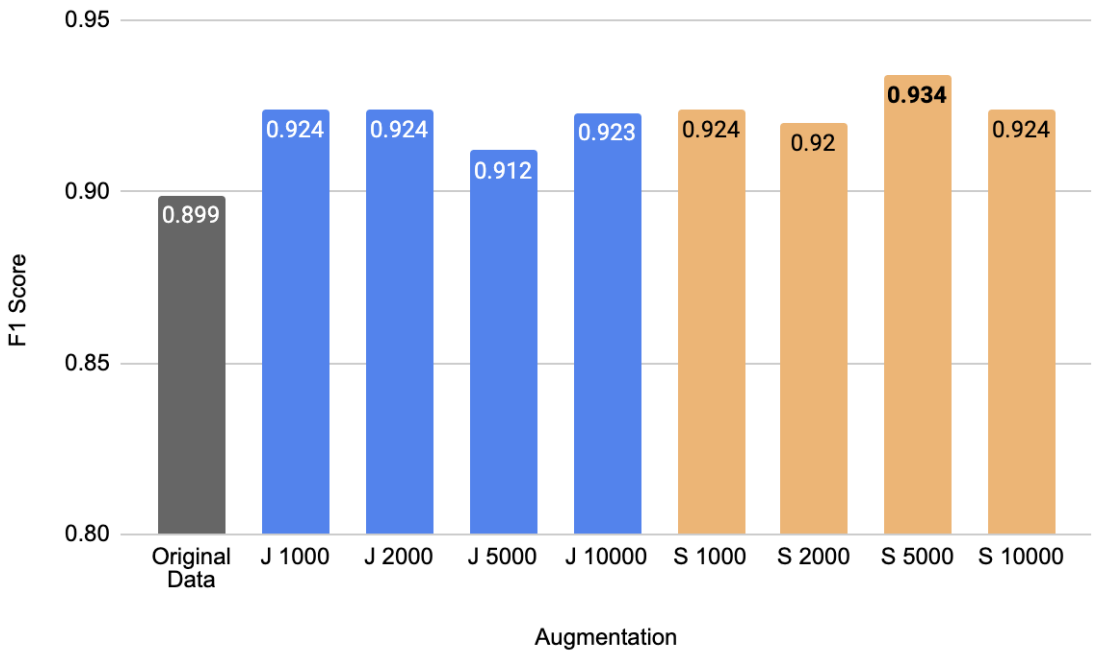

Figure 6. Hyperparameter Optimization을 진행한 LSTM 모델에 대해서 Data Augmentation을 적용하였을 때의 F1 Score.

J는 Jittering, S는 Scaling으로 Data Augmentation을 적용한 데이터를 의미한다. Data Augmentation을 하지 않은 경우(0.899)와 비교했을 때보다 적용했을 때 다소 높은 F1 Score를 갖는 것으로 나타났다. 가장 성능이 높은 경우는 Scaling 방식으로 5000개를 생성하여 Augment한 데이터였다(0.934).

Feature Selection

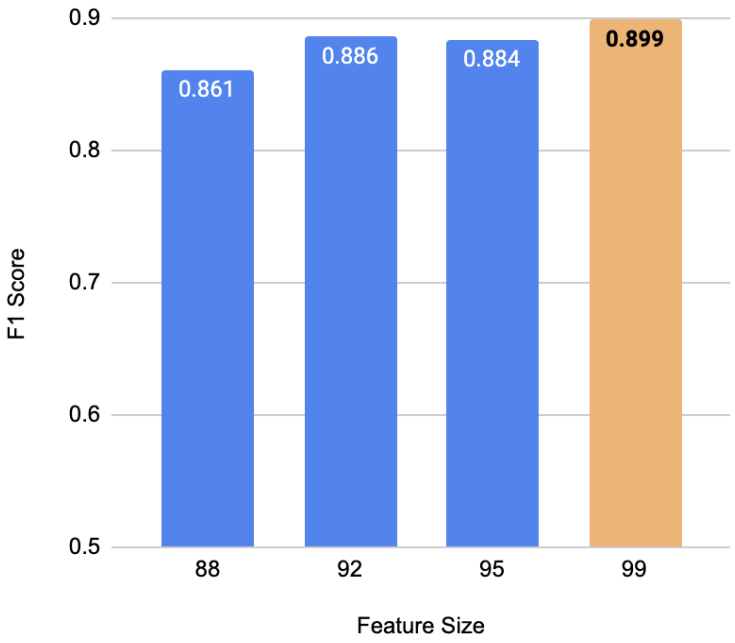

Figure 7. Hyperparameter Optimization을 진행한 LSTM 모델에 대해 Feature Selection을 적용하였을 때의 F1 Score.

Feature Selection을 진행했을 때, 전체 Feature Set(0.899)와 비교했을 때와 비교해서 더 낮은 F1 Score을 보였다.

Ensemble

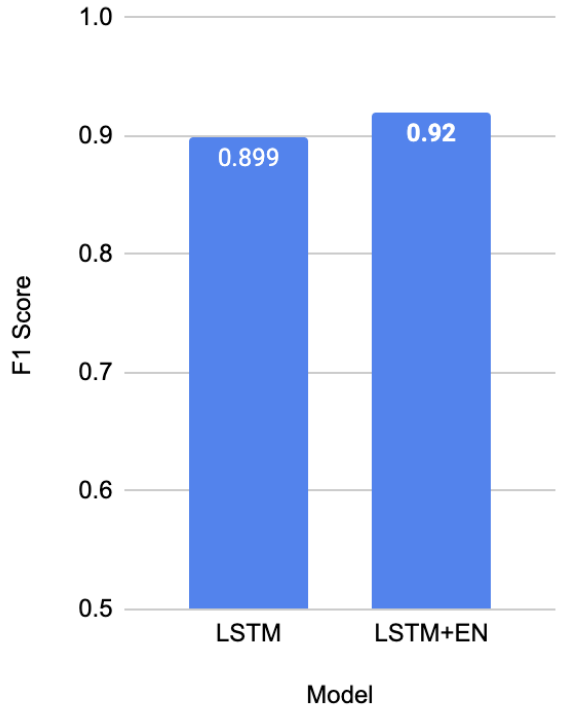

Figure 8. Hyperparameter Optimization을 진행한 LSTM 모델에 대해 Ensemble을 적용하였을 때의 F1 Score.

Data Augmentation을 적용한 모델에 추가로 Majority Voting Ensemble(n=5)을 적용한 결과, F1 Score가 기존 0.899에서 0.92로 증가하였다.

종합

Random Classifier(RC), Random Forest(RF), LSTM, 총 세 가지 모델을 비교하였다. LSTM의 경우 Data Augmentation, Feature Selection, Ensemble을 적용한 모델을 추가로 비교했다. Random Classifier는 12%를 PDA로, 88%를 Control로 임의로 분류하는 모델을 가정하였다. Random Forest는 기존 연구의 결과를 참조했다. 마지막으로, 좋은 성능을 보인 Data Augmentation과 Ensemble을 동시에 적용한 모델을 비교하였다. LSTM+DA+EN는 최적의 Hyperparameter를 가진

LSTM에 대해 Scaling 방식으로 5000개 데이터를 Augment하여 학습한 모델을 사용하였고, 추가로 Ensemble을 적용한 모델이다.

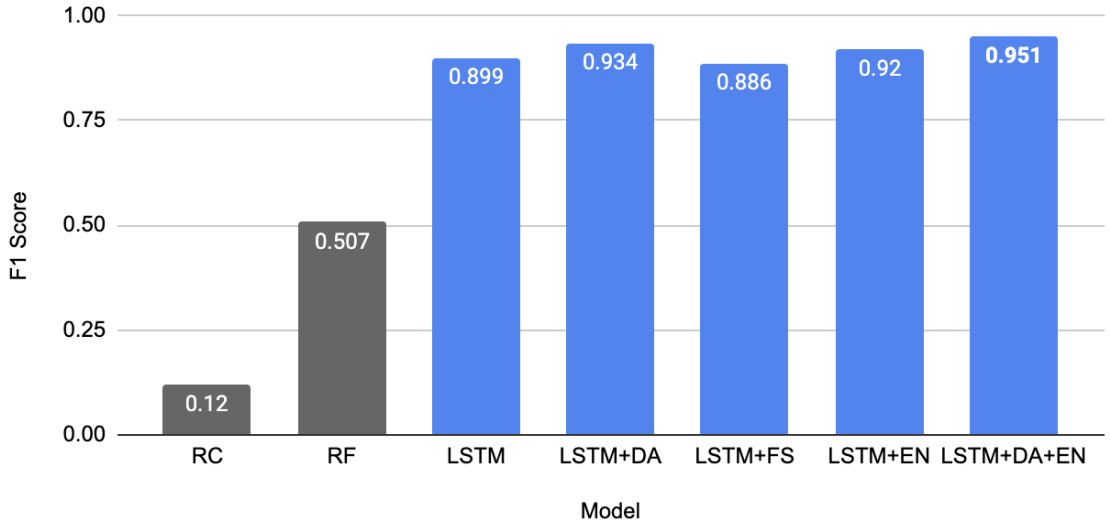

Figure 9. 전체 모델에 대해서 F1 Score를 비교한 그래프.

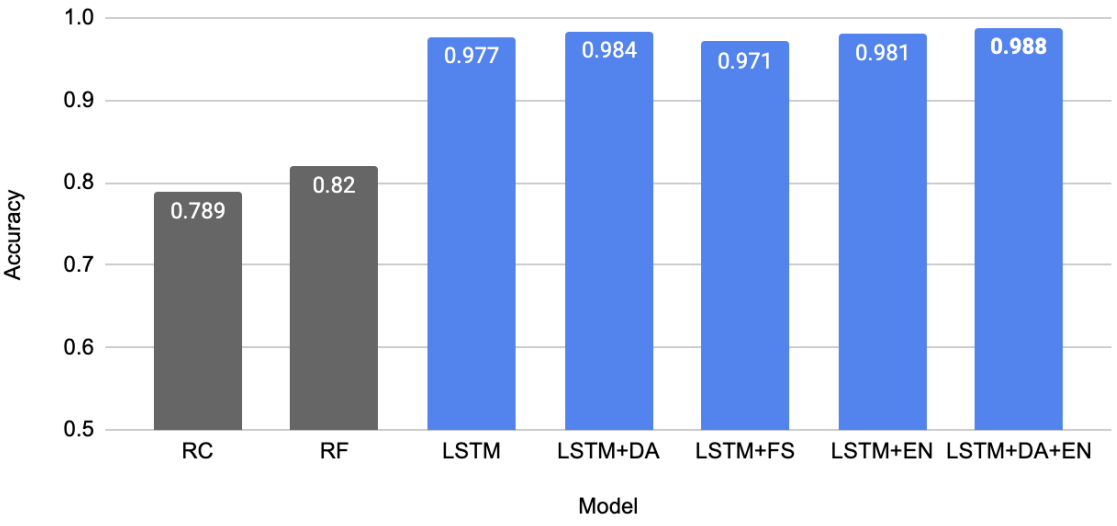

Figure 10. 전체 모델에 대해서 Accuracy를 비교한 그래프.

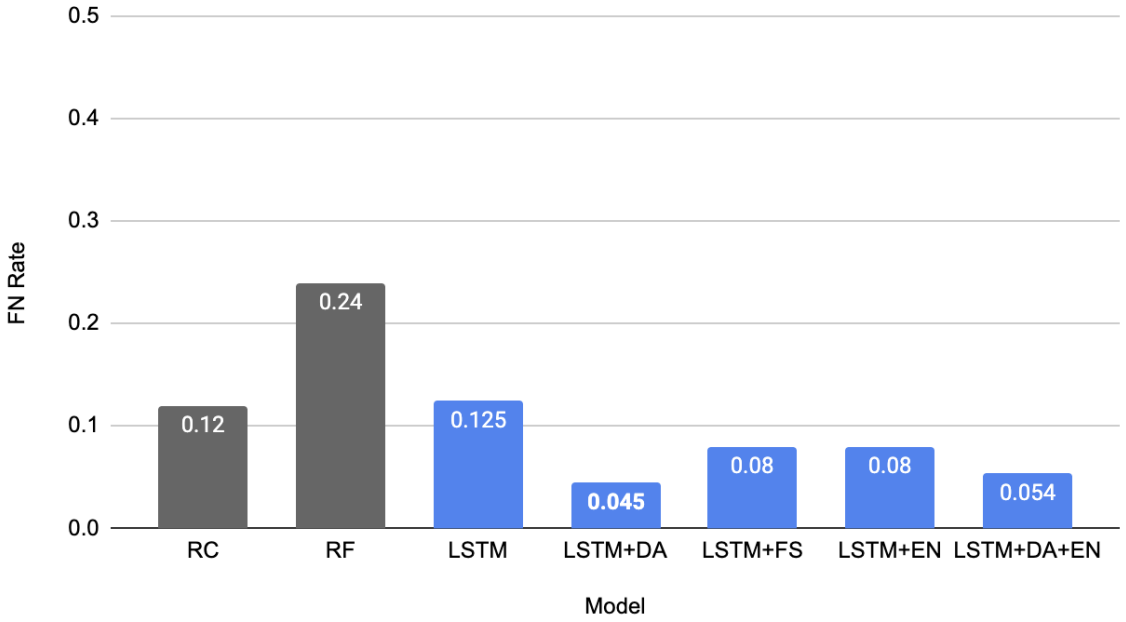

Figure 11. 전체 모델에 대해서 FN Rate를 비교한 그래프.

RC, RF와 비교하였을 때 본 연구에서 제시한 모델(LSTM+DA+EN)은 F1 Score(0.951)와 Accuracy(0.988)는 큰 차이로 높게 나타나고 FN Rate(0.054)는 큰 차이로 낮게 나타났다. RF와 비교했을 때 전반적으로 제시한 모델이 PDA 여부를 더 정확하게 예측하는 것을 알 수 있다.

FN rate은 양성을 음성으로 잘못 판단하는 Error의 비율이다. 의료 데이터 모델에서 중요하게 봐야할 지표로 낮은 값을 달성할수록 좋다. FN Rate는 LSTM과 Data Augmentation을 혼합 적용했을 때 가장 낮은 값(0.045)을 보였다.

Conclusion

LSTM, Data Augmentation, Ensemble을 혼합 적용한 모델이 전반적으로 높은 성능을 보였다. 적은 데이터셋에 딥러닝을 적용하기 위해 Data augmentation, Ensemble 등의 기법을 시도하였고, 확실히 향상된 성능을 보일 수 있었다. 이러한 시도를 통해 PDA와 마찬가지로 다른 다양한 희귀질환의 예측에도 이와 같은 방법을 시도해볼 수 있다고 볼 수 있다.

Discussion

Feature Selection의 경우 큰 성능 향상을 이루지 못했다. 그 원인으로는 Feature Importance를 얻어내기 위해 사용한 Random Forest 모델의 성능(F1 Score = 0.48)이 RNN 기반 모델(F1 Score = 0.899)과 비교했을 때 낮기 때문으로 생각된다. 이 문제를 해결하기 위해서는 RNN 기반 모델을 직접적으로 사용하는 Permutation Feature Importance 등을 통한 후속 연구가 이어질 수 있을 것이다.

medGAN을 사용한 Data Augmentation의 경우 필요한 성능이 나오지 않아서 결과에서 제외하였다. 실제 데이터에 적용하여 학습을 진행한 결과, Generator가 충분히 주어진 데이터의 분포를 학습하기 전에 Local Minimum에 빠져버리는 것을 확인할 수 있었다. 이를 해결하고자 Hyperparameter의 조정 등을 해결책을 시도했으나, 개선되지 않았다.

Dataset	(A) Sutter PAMF	(B) MIMIC-III	(C) Sutter Heart Failure
# of patients	258,559	46,520	30,738
# of unique codes	615	1071	569
Avg. # of codes per patient	38.37	11.27	53.02
Max # of codes for a patient	198	90	871
Min # of codes for a patient	1	1	2

이 결과는 절대적인 데이터 수의 한계인 것으로 보인다. 위에서 볼 수 있듯 medGAN 논문에서 환자의 수는 최소 3만 여 명을 대상으로 했으나, 본 연구는 4백여 명을 대상으로 Augmentation을 시도하기 때문에 보다 많은 데이터를 요하는 GAN 방식에 적합하지 않았던 것으로 보인다.

Acknowledgement

본 연구는 연세대학교 디펜더블 컴퓨팅 연구실의 [redacted] 연구원의 연구에 바탕을 두어 진행되었다. 학습에 사용된 데이터셋은 연세대학교 세브란스로부터 제공 받은 환자의 의료 정보를 바탕으로 얻어졌다. 해당 데이터는 [redacted] 연구원이 Pre-processing을 진행하였고, 이후 개인정보 보호를 위해 재가공되어 제공되었다.

역할 분담

[redacted]

- medGAN
- Feature Selection
- Hyperparameter Optimization

[redacted]

- 팀장
- Preprocessing
- Random transformation-based data augmentation
- Feature Selection
- Ensemble

[redacted]

- RNN Models
- Data Visualization
- Presentation

Reference

[1] Roohallah Alizadehsani, Moloud Abdar, Mohamad Roshanzamir, Abbas Khosravi, Parham M. Kebria, Fahime Khozeimeh, Saeid Nahavandi, Nizal Sarrafzadegan, U. Rajendra Acharya. “Machine learning-based coronary artery disease diagnosis: A comprehensive review” (2019). Computers in Biology and Medicine. Volume 111.

[2] Nisreen I.R. Yassin, Shaimaa Omran, Enas M.F. El Houby, Hemat Allam. “Machine learning techniques for breast cancer computer aided diagnosis using different image modalities: A

systematic review" (2018). Computer Methods and Programs in Biomedicine. Volume 156. Pages 25-45.

[3] Bhavsar, Kaustubh Arun; Abugabah, Ahed; Singla, Jimmy; AlZubi, Ahmad Ali; Bashir, Ali Kashif; and Nikita. "A comprehensive review on medical diagnosis using machine learning" (2021). All Works. 4101.

[4] Iwana BK, Uchida S. "An empirical survey of data augmentation for time series classification with neural networks" (2021). PLOS ONE. 16(7).

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. "Generative Adversarial Networks" (2014).

[6] Edward Choi, Siddharth Biswal, Bradley Malin, Jon Duke, Walter F. Stewart, Jimeng Sun. "Generating Multi-label Discrete Patient Records using Generative Adversarial Networks" (2017). Proceedings of Machine Learning for Healthcare 2017.

[7] Weiwei Huo, Weier Li, Zehui Zhang, Chao Sun, Feikun Zhou, Guoqing Gong. "Performance prediction of proton-exchange membrane fuel cell based on convolutional neural network and random forest feature selection" (2021). Energy Conversion and Management. Volume 243.

[8] TAYLOR MAULDIN, ANNE H. NGU, VANGELIS METSIS, MARC E. CANBY. "Ensemble Deep Learning on Wearables Using Small Datasets" (2020). ACM Transactions on Computing for Healthcare, Vol. 2, No. 1, Article 5.

[9] Rahma Atallah, Amjed Al-Mousa. "Heart Disease Detection Using Machine Learning Majority Voting Ensemble Method" (2019). Conference on new trends in computing.

[10] Sepp Hochreiter; Jürgen Schmidhuber. "Long Short-Term Memory" (1997). Neural Computation. Volume 9, Issue 8.

[11] Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, Yoshua Bengio. "On the Properties of Neural Machine Translation: Encoder-Decoder Approaches" (2014).

[12] Laila Rasmy, Yonghui Wu, Ningtao Wang, Xin Geng, W. Jim Zheng, Fei Wang, Hulin Wu, Hua Xu, Degui Zhi. "A study of generalizability of recurrent neural network-based predictive models for heart failure onset risk using a large and heterogeneous EHR data set" (2018). Journal of Biomedical Informatics, Volume 84.

[13] Liaw, Richard and Liang, Eric and Nishihara, Robert and Moritz, Philipp and Gonzalez, Joseph E and Stoica, Ion. “Tune: A Research Platform for Distributed Model Selection and Training” (2018). ICML AutoML workshop.