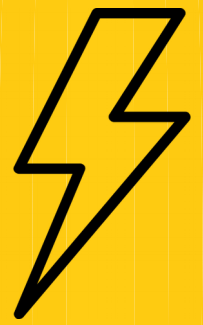


스팀터빈 사전이상감지
인공지능 설계 및 구현

A Table of Contents


PART 1

서론

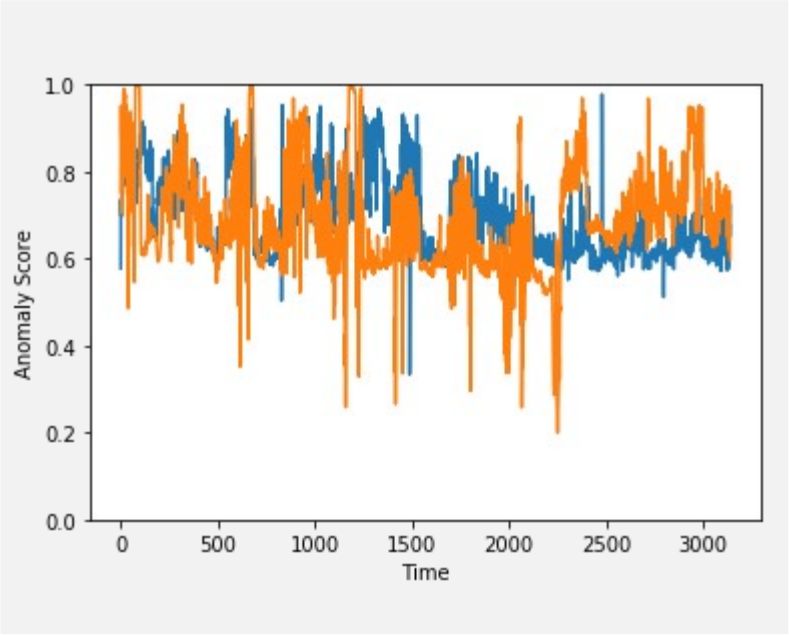
가. 과제 요약

나. 선행 연구 사례

PART 1

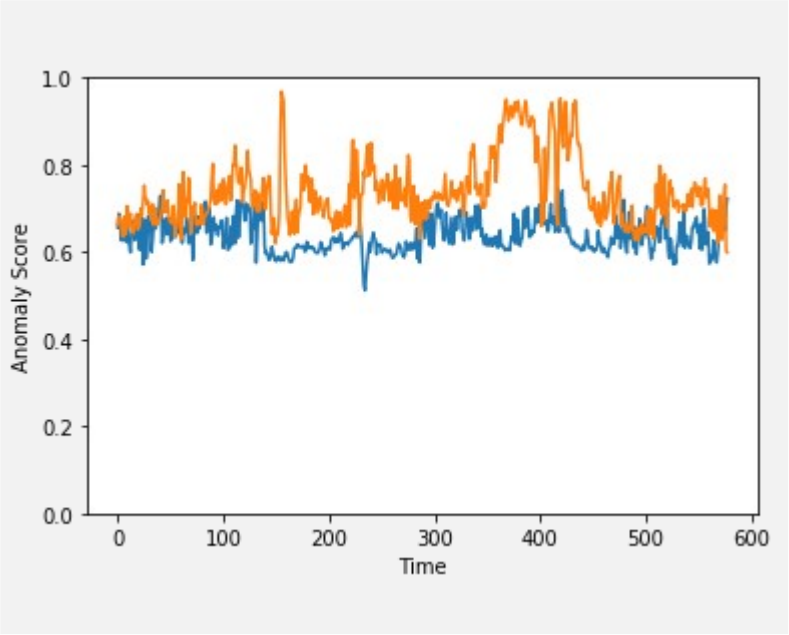
과제의 요약

기존에 미리 예측하지 못했던 순간적인 스팀터빈의 고장을 사전에 감지할 수 있는 인공지능의 설계와 구현



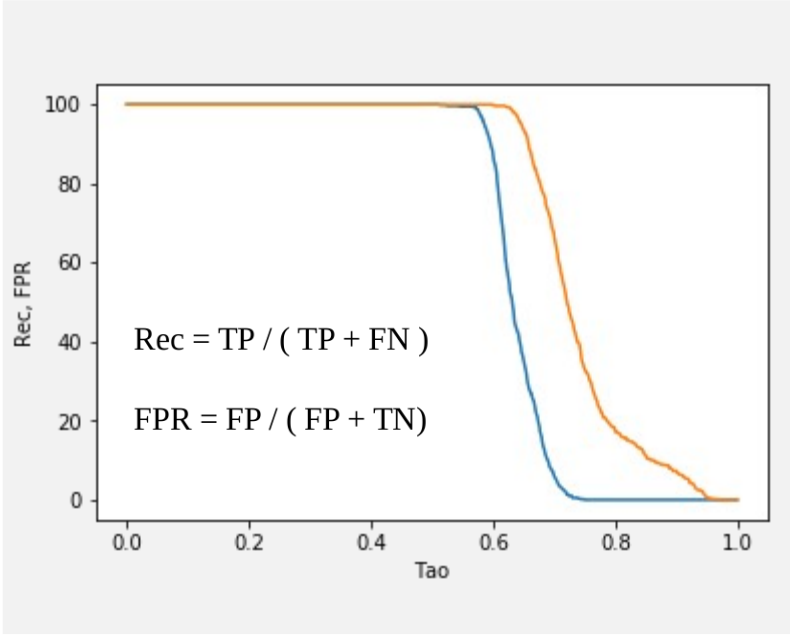
고장 시점 전 11일 동안

처음 8일 동안은 정상 데이터와 고장 데이터가 비슷한 수준에서 다소 우하향하는 트렌드를 보인다. 그러나 고장 3일 정도 전부터 계속 트렌드를 유지하는 정상 데이터와 달리 고장 데이터에서는 Anomaly Score가 정상 데이터보다 유의미하게 높은 수치를 보인다.



고장 시점 전 2일 동안

고장 직전 2일 동안의 Anomaly Score만 비교하면 더욱 확연하게 정상데이터와 고장데이터가 구분된다. 본 과제의 모델을 통해 고장의 징후를 감지할 수 있는 것을 확인하였다.



Tao에 따른 Rec와 FPR

고장 시점 2일 전의 데이터에 대해 0부터 1까지 이상 감지 기준(이하 Tao)을 바꾸어가며 Rec와 FPR을 비교한 것이다. 이 둘은 Tao에 따른 trade-off가 존재하여 적절한 Tao를 설정하는 것이 중요하다. 해당 고장 사례에 대해서 최적으로 판단되는 0.7의 Tao 하에서 약 66.15%의 Rec와 약 5.70%의 FPR이 관찰되었다.

PART 1

선행 연구 사례

딥러닝으로 스팀 터빈의 이상을 감지하는 세 가지 연구 사례

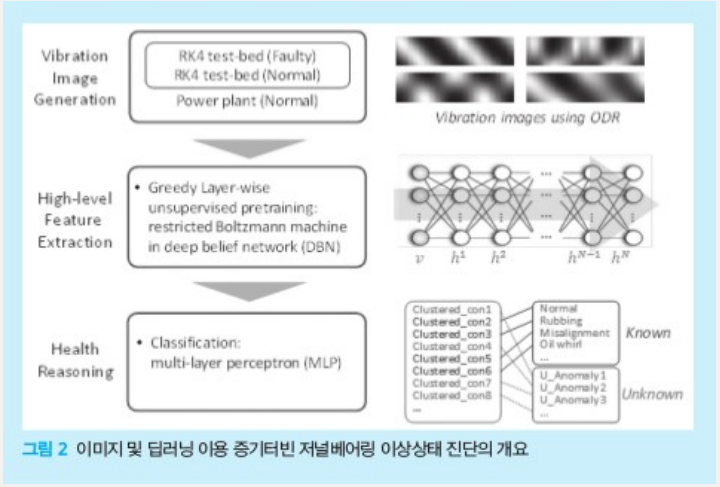


그림 2 이미지 및 딥러닝 이용 증기터빈 저널베어링 이상상태 진단의 개요

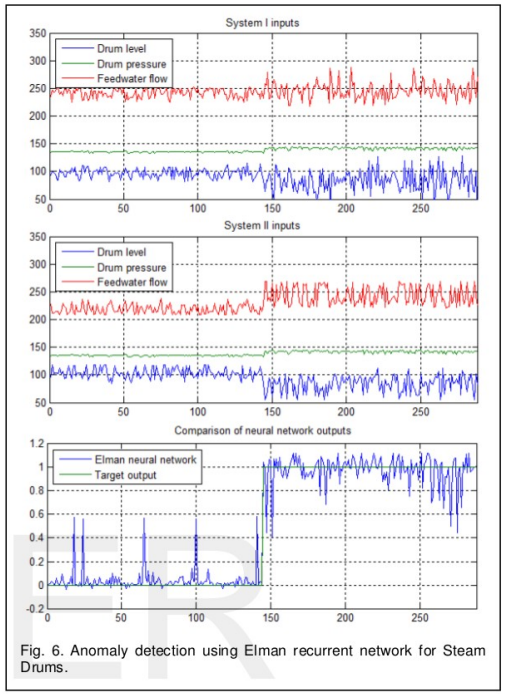
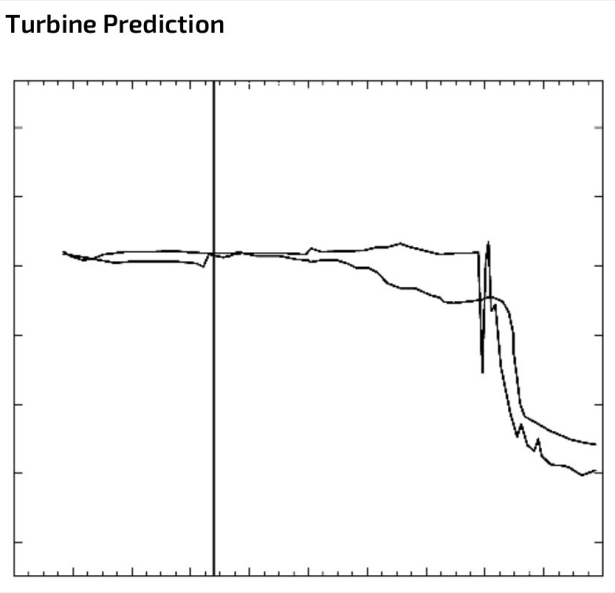


Fig. 6. Anomaly detection using Elman recurrent network for Steam Drums.



Smart diagnosis of journal bearing

이 연구는 스팀터빈의 진동 데이터를 이미지로 변환하여 딥러닝 네트워크의 일종인 CNN을 사용하였다는 점에서 본 과제의 모델과 차이가 있지만 딥러닝으로 스팀 터빈의 이상을 감지했다는 점과 고장 데이터를 시뮬레이션으로 생성하였다는 점에서 참고가 된다.

AD-in-Thermal-Power-Plant

2015년에 수행된 이 연구는 스팀 터빈은 아니지만 Steam Drum과 Steam Superheater에 대한 이상 감지를 수행하였다. 인풋 데이터가 시계열 데이터라는 점에서 순서가 있는 데이터의 처리에 용이한 딥러닝 모델의 일종인 RNN을 사용했다는 점에서 본 과제에 참고가 되었다.

Prediction of Turbine Failure

이 연구를 통해 스팀 터빈의 갑작스러운 고장을 예측하는 것이 가능하다는 것을 확인할 수 있었으나 구체적으로 어떠한 데이터를 사용하였고 어떤 모델로 설계되었는지 파악할 수 없었다. 또한 이 연구의 모델 또한 이상은 감지하지만 그 이상이 구체적으로 blade tear라는 것을 보여주지 못한다는 점에서 본 과제와 유사하다.

PART 2

과제 수행

ㄱ. 데이터셋의 분할

ㄴ. 데이터 차원 축소

ㄷ. LSTM

ㄹ. GAN

ㅁ. MAD-GAN

PART 2

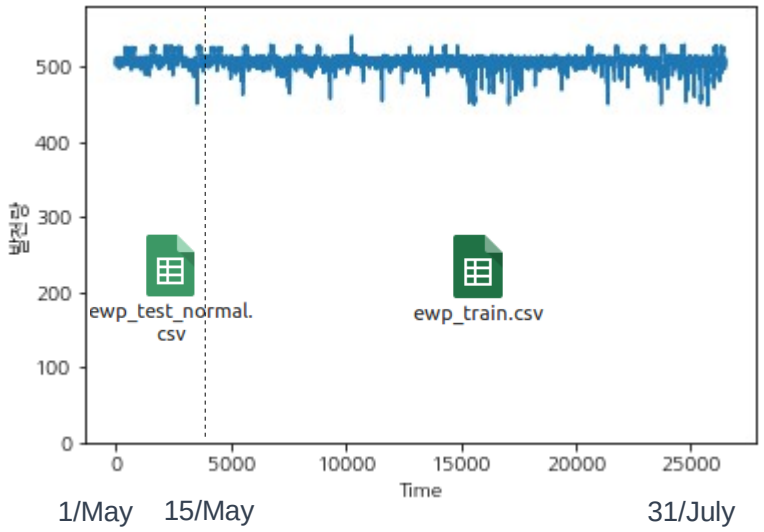
데이터셋의 분할

한국동서발전으로부터 제공 받은 스팀터빈 운전 데이터를 분할하여
학습 데이터와 테스트 데이터를 구성

01

정상 데이터

2000년 5월 1일부터 2000년 7월 31일까지 정상 운행한 스팀 터빈으로부터 약 5분 간격으로 생성된 운전 데이터



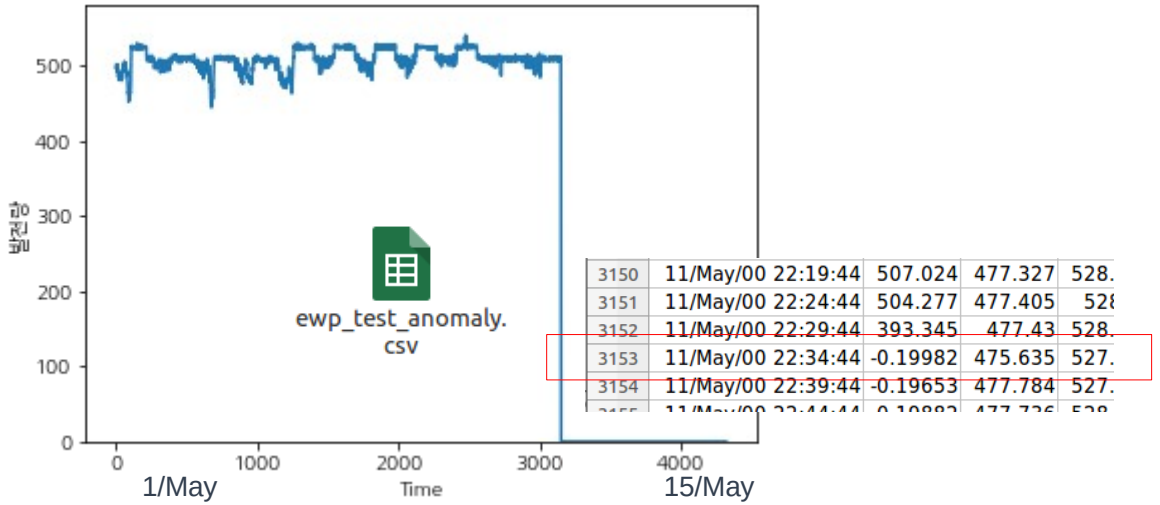
1/May~ 15/May : test data for normal

16/May~31/July : train data

02

고장 데이터

2000년 5월 1일부터 2000년 5월 15일까지 고장이 발생한 스팀 터빈으로부터 약 5분 간격으로 생성된 운전 데이터



1/May~ 11/May 22:24(just before failure) : test data for anomaly

PART 2

데이터 차원 축소

PCA를 수행하여 398차원의 데이터를 6차원으로 축소

01

Raw data

발전량, 온도, 진동, 압력 등의 398개의 변수로 구성된다. 딥러닝의 연산량은 데이터의 차원에 따라 지수적으로 증가한다. 또한 약 22만 7천 개의 학습데이터 크기를 고려했을 때 398 차원은 학습이 효과적으로 이루어지기에 너무 크다.

0	발전량	23DH-MW
1	HP TBN ROTOR BORE TEMP	23MK-ROTOR_1_TEMP
2	LP TBN INLET ROTOR BORE TEMP	23MK-ROTOR_2_TEMP
3	LP TBN OUTLET ROTOR BORE TEMP	23MK-ROTOR_3_TEMP
4	1ST STAGE SHELL LOSER INNER TEMP	23MK-TT_FSS
5	REHEAT BOWL INNER SURFACE TEMP	23MK-RT_RBS
6	CROSSOVER CHAMBER UPPER TEMP	23MK-TT_XOU
7	Aux Header Press	23MK-AP_ASSS
8	Main Steam Press	23MK-FP_MSP_PSI

. . .

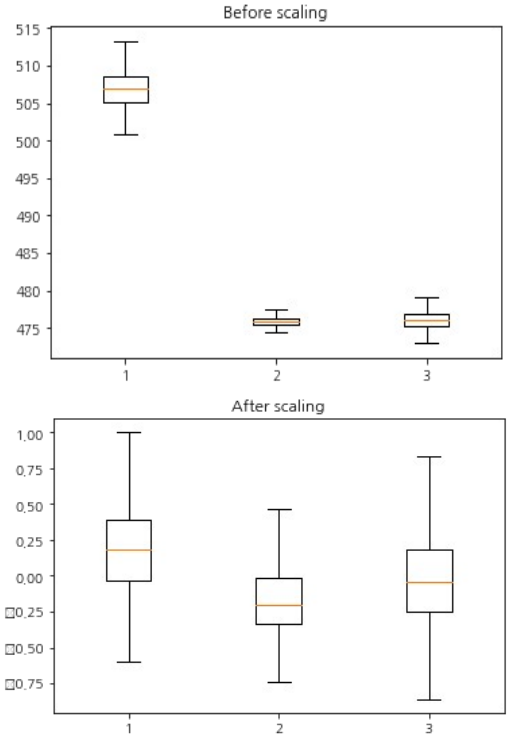
391	Cross Over STM Temp	23PI_TCO
392	Cold RH STM Pr	23PI_PCRS
393	Cold RH STM Temp	23PI_TCRS
394	Hot RH STM Pr	23PI_PHRS
395	Hot RH STM Temp	23PI_THRS
396	Main STM Pr	23PI_PMS
397	Main STM Temp	23PI_TMS

398 feature

02

Scaling

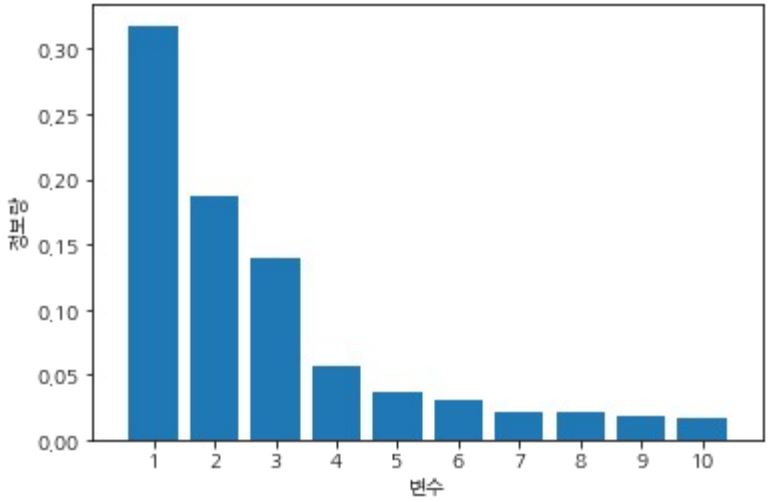
PCA를 수행할 때 단위가 큰 변수는 그만큼 영향도 커진다. 각 변수의 영향력을 같게 하기 위해서 단위를 -1에서 1로 통일하였다.



03

PCA(주성분분석)

약 3.1% 정도의 정보량을 갖는 6번째 변수 이후 7번째 변수부터 약 2% 내외의 정보량을 가진다. 1번째 변수부터 6번째 변수가 갖는 정보량은 약 76.98%로서 연산의 효율성과 결과의 신뢰성을 고려하였을 때 축소된 차원의 수는 6차원이 적절하다고 판단하였다.

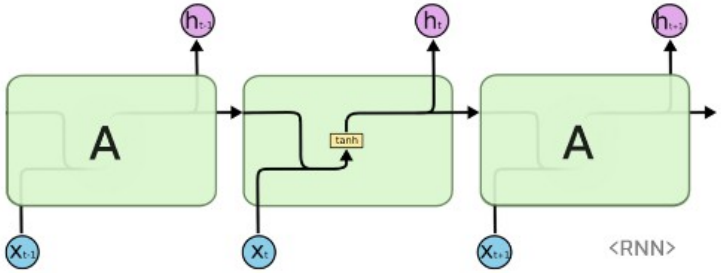


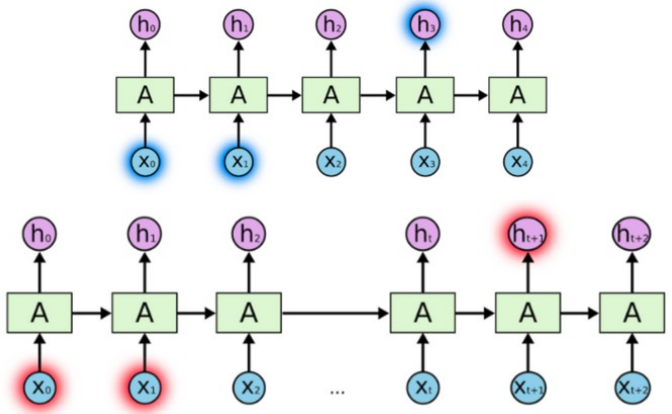
PCA 변수에 따른 정보량

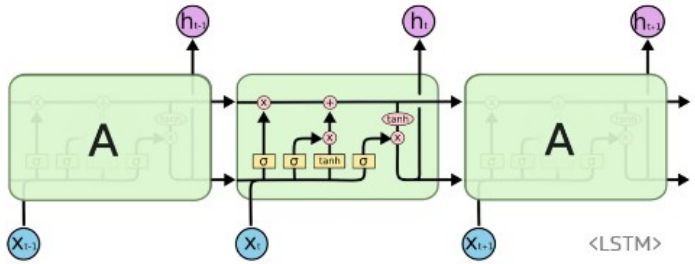
PART 2

LSTM

Long Short-Term Memory models







RNN

RNN은 위 그림과 같이 히든 노드가 방향을 가지는 엣지로 연결돼 순환구조를 이루는 인공신경망의 한 종류이다. 이러한 구조 때문에 음성, 문자 등 순차적으로 등장하는 데이터(시퀀스) 처리에 적합한 모델로 알려져 있는데 본 과제도 5분 간격으로 생성되는 데이터가 순차적으로 등장한다는 점에서 RNN은 적합하다.

vanishing gradient problem

RNN은 위 그림의 아래와 같이 관련 정보와 그 정보를 사용하는 지점 사이 거리가 멀 경우 학습 능력이 저하되는데 이를 vanishing gradient problem이라 한다.

LSTM

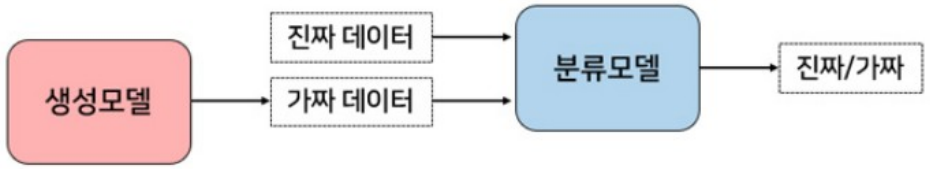
LSTM은 RNN의 히든 스테이트에 cell-state를 추가하여 이 문제의 크기를 줄였다. 그림2.4에서 두 노드 사이를 연결하는 방향을 가지는 두 엣지 중에 위의 엣지가 바로 cell-state인데 이 cell-state가 일종의 컨베이어 벨트 역할을 하여 거리가 멀어지더라도 그래디언트가 비교적 잘 전파된다.

PART 2

GAN

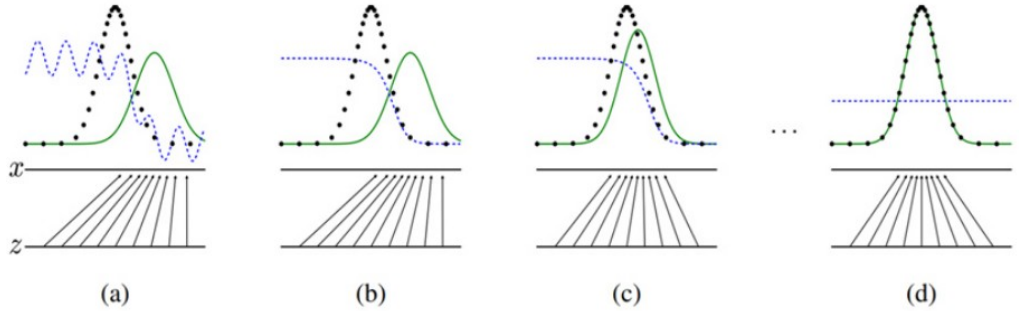
Generative Adversarial Networks

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[log\ D(x)] + E_{z\sim p_z(z)}[log(1-D(G(z))]$$



GAN의 구조

Ian Goodfellow는 이 두 모델을 경찰과 지폐 위조범의 대립으로 비유하였다. 위조지폐범은 최대한 진짜 같은 화폐를 만들어(생성) 경찰을 속이기 위해 노력하고, 경찰은 진짜 화폐와 가짜 화페를 완벽히 판별(분류)하여 위조지폐범을 검가하는 것을 목표로 한다. 이러한 경쟁적인 학습이 지속되다 보면 어느 순간 위조지폐범은 진짜와 다를 바 없는 위조지폐를 만들 수 있게 된다.



※ 검은 점선: 원 데이터의 확률분포, 녹색 점선: GAN이 만들어 내는 확률분포, 파란 점선: 분류자의 확률분포

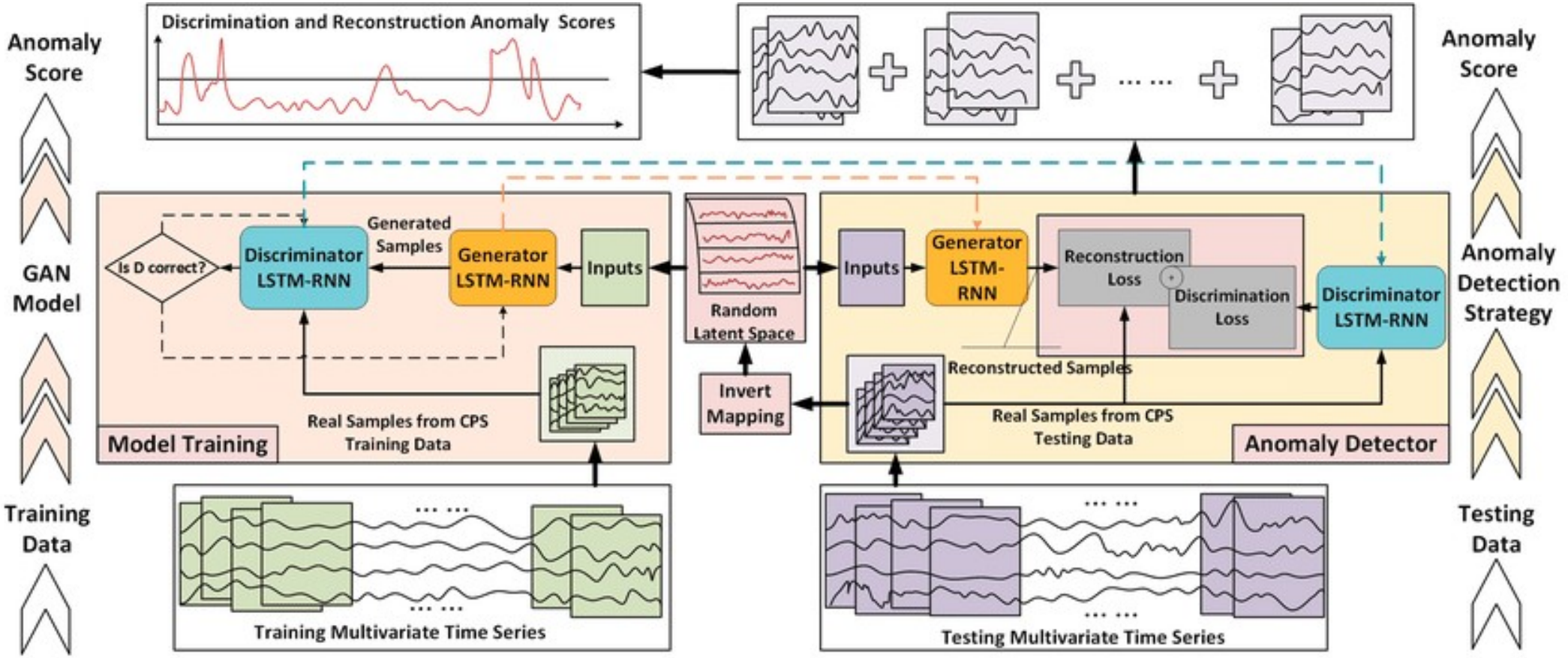
생성자 G의 성질

생성자 G가 학습한 딥러닝 매핑(Z->G(Z))이 단순히 불연속적인 1:1 매칭이 아니라 실제 데이터의 확률분포를 정확히 표현하고 있어서, 입력에서의 약간의 변화는 출력에서도 부드러운 변화로 표현이 가능하다는 것이 증명되었다(the smooth transitions of latent space).

PART 2

MAD-GAN

Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks




PART 3

결과 분석

ㄱ. 최적 epoch 및 모델 탐색

ㄴ. 시기별 이상 감지 결과

PART 3

최적 epoch과 모델 탐색

epoch499에서 판별자 D만 사용한 결과가 가장 우수

01 판별자 D만 사용한 결과

epoch499에서 판별자 D만 사용한 결과가 확연히 우수한 것을 확인할 수 있다.

Epoch	N(정답 개수)	Accu (%)	Rec (%)	FPR (%)
99	161	31.6	27.09	62.55
299	340	66.8	37.85	1.992
499	490	96.27	99.2	3.586
599	363	71.62	54.98	9.96
699	411	80.75	96.41	32.27

02 생성자 G만 사용한 결과

생성자 G도 epoch499에서는 유의미한 이상감지 성능을 보였지만 판별자 D에는 미치지 못하였다.

Epoch	N(정답 개수)	Accu (%)	Rec (%)	FPR (%)
99	238	46.76	6.773	11.55
299	258	50.69	39.04	35.86
499	325	63.85	49.8	19.92
599	248	48.72	11.55	12.35
699	313	61.49	64.14	39.04

03 두 모듈을 다 사용한 결과

이 둘을 함께 사용하면 오히려 판별자 D를 단독으로 사용할 때보다 성능이 감소하여 생성자 G는 판별자 D를 보완하지도 못하는 것으로 보인다.

Epoch	N(정답 개수)	Accu (%)	Rec (%)	FPR (%)
99	256	50.29	2.39	0.0
299	260	51.08	3.984	0.0
499	475	93.32	92.83	3.187
599	332	65.23	38.65	5.976
699	361	70.92	54.58	10.36

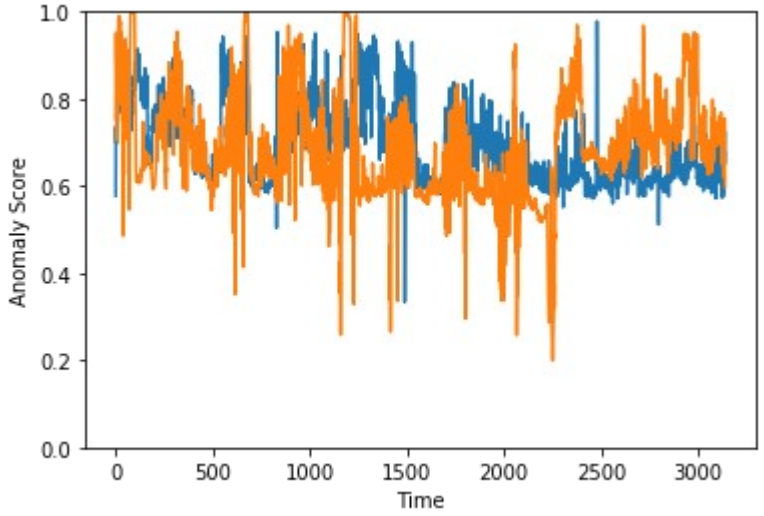
PART 3

시기별 이상 감지 결과

3일 정도 전부터 이상 징후를 감지. 고장이 가까워질수록 이상 징후는 강해진다.

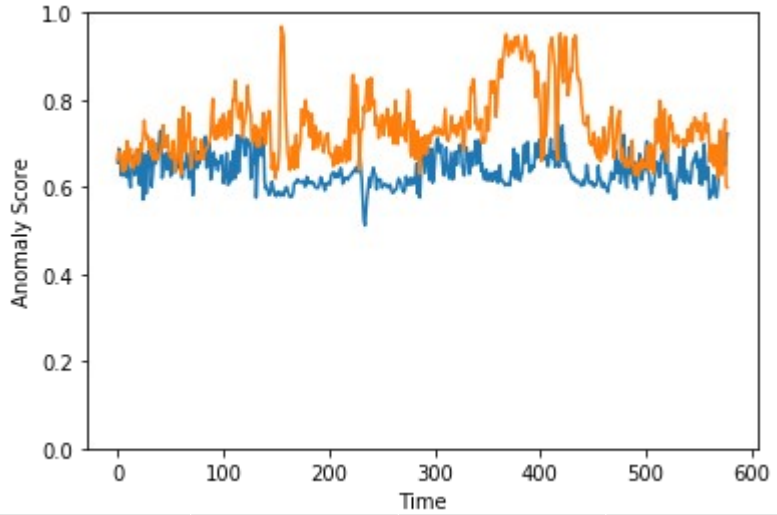
01 고장 전 11일 동안 이상값

고장 3일전까지는 비슷한 수준을 보이다가 3일 전부터 고장 데이터의 이상값이 높다.



02 고장 전 2일 동안 이상값

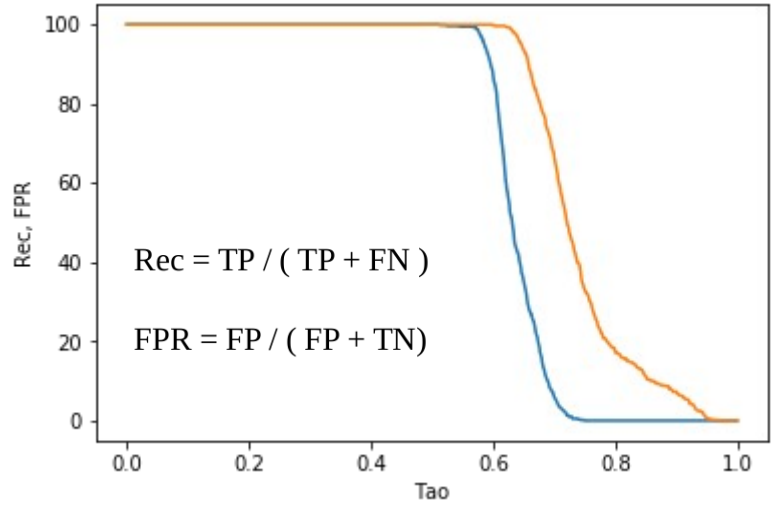
정상 데이터에 비해 고장 데이터의 이상값이 높은 것을 확인할 수 있다.



D-day	Accu	Rec	FPR
1	82.28	72.76	5.862
2	75.81	59.31	5.517
3	68.99	46.55	6.552

03 기준에 따른 이상 감지

해당 고장 사례에 대해서 최적으로 판단되는 0.7의 Tao 에서 약 66.15%의 Rec와 약 5.70%의 FPR이 관찰되었다.




PART 4

한계점 및 향후 과제

가. 고장에 대한 구체적인 정보 부재

본 과제는 고장의 징후를 미리 감지해서 알려줄 뿐 어디에서 어떤 문제가 발생할 수 있는지는 전혀 알려주지 않는다. 본 과제의 인공지능을 통해 고장의 징후를 감지했을때 구체적인 진단을 내릴 수 있는 인공지능을 설계하는 것은 향후의 과제가 될 것이다.

나. 계절성의 존재

본 과제는 학습에 사용된 데이터가 생성된 시기와 테스트에 사용된 데이터가 생성된 시기가 다르다. 이상값이 시간에 따라 우하향하는 경향을 보이는 것도 이러한 원인이 있지 않을까 추측한다. 학습 데이터가 생성된 시기에서 멀어질수록 인공지능은 정상에서 멀어진다고 판단하는 것이다. 이 추측과는 별개로 계절성을 고려하여 데이터셋을 구축하고 인공지능을 설계한다면 성능을 더욱 향상시킬 여지가 있는 것은 분명하다.

다. 특정 고장 사례에 과적합

본 과제의 인공지능의 검증과 세팅에 사용한 고장 사례는 하나이다. 본 과제에서는 Tao, epoch 등 여러 세팅 값이 존재한다. 이 값은 하나의 고장 사례를 보며 최적화를 하였기 때문에 일반적인 스팀터빈의 고장 사례에서는 최적이 아닐 수 있다. 이 문제는 여러 고장 사례에 대해서 검증함으로써 극복할 수 있다.

참고문헌

[1] Oh, Hyunseok, et al.
“Smart diagnosis of journal bearing rotor systems: Unsupervised feature extraction scheme by deep learning.” (2016).

[2] A. Hajdarevic, et al.
”Recurent-Neural-Network-as-a-Tool-for-Parameter-Anomaly-Detection-in-Thermal-Power-Plant” (2015).

[3] Dr. Patrick Bangert, “Prediction of Turbine Failure”.

[4] “RNN과 LSTM을 이해해보자!”, https://ratsgo.github.io/natural%20language%20processing/2017/03/09/rnnlstm/

[5] 장준혁, “GAN의 활용 사례와 발전방향”,
https://www.samsungsds.com/global/ko/support/insights/Generative-adversarial-network-AI-3.html

[6] Dan Li, et al.
“MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks”(2019).

[7] R. Alec, L. Metz, and S. Chintala,
“Unsupervised representation learning with deep convolutional generative adversarial networks,” (2015).

THANK YOU