Journal of the Korean Institute of Industrial Engineers
Vol. 49, No. 3, pp. 267-275, June 2023.
ISSN 1225-0988 | EISSN 2234-6457
https://doi.org/10.7232/JKIIE.2023.49.3.267

MADFlow : Normalizing Flow를 활용한 다변량 시계열 이상 탐지

문지원 · 송승환 · 백준걸†

고려대학교 산업경영공학과

MADFlow : Multivariate Time Series Anomaly Detection via Normalizing Flow

Jiwon Moon · Seunghwan Song · Jun-Geol Baek

Department of Industrial and Management Engineering, Korea University

With the recent advancement of smart factories in manufacturing processes, high-dimensional data is being collected in real-time from multiple sensors in production facilities. However, it is very difficult to detect anomalies that reflect both correlations and temporal dependency between high-dimensional variables. In this study, we propose Multivariate Time Series Anomaly Detection via Normalizing Flow (MADFlow), which can reflect both correlation between variables and temporal dependency. MADFlow consists of a temporal encoder to reflect temporal dependency and a flow module to learn the distribution of high-dimensional data and is trained in an end-to-end manner. Experimental results on multivariate time series data with similar characteristics to data generated in manufacturing processes show that MADFlow has significantly better anomaly detection performance than existing models. Therefore, we expect MADFlow to be able to efficiently detect anomalies in real-world manufacturing processes.

Keywords: Anomaly Detection, Normalizing Flow, Masked Autoregressive Flow, Real-Values Non-Volume Preserving, Long Short Term Memory

1. 서 론

확률적으로 정의되는 이상(anomaly)이란 정규성의 개념에서 크게 벗어나는 관측치들을 의미한다. 즉, 이상을 탐지한다는 것은 대부분의 데이터로부터 멀리 떨어진 비정상적인 관측치를 식별하는 연구 분야이다(Ruff *et al.*, 2021). 이상 탐지(anomaly detection)는 제조 공정, 카드 사기, 의료 등 다양한 도메인에서 적용되고 있다. 특히, 제조 공정에서의 이상 탐지는 실제 산업 현장의 생산 수율과 직결된다. 따라서 신속하고 정확한 이상 탐지가 이루어져야 작업 중단과 품질 저하로부터 발생할 수 있는 막대한 경제적 손실을 예방할 수 있다.

최근 제조 공정은 스마트 팩토리(smart factory)의 영향으로 생산 시스템이 고도화됨에 따라 설비의 여러 센서들로부터 데이터가 실시간으로 수집되고 있다. 그러나 다수의 센서들로부터 수집되는 다변량 시계열 데이터에 대한 이상 탐지는 다음과 같은 이유로 어려움을 겪는다. 첫째, 고차원 변수 간의 상관관계(correlation) 뿐만 아니라 각각의 시계열 데이터의 복잡한 시간 의존성(temporal dependency)까지 반영해야 한다. 제조 공정에 사용된 대부분의 기존 방법들은 센서들 간의 상관관계와 시간 상의 의존관계를 제대로 학습하지 못한다. 두 가지 특성

본 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구(NRF-2022R1A2C2004457)이며 교육부 및 한국연구재단의 4단계 BK21 사업으로 지원된 연구임. 또한, 삼성전자(Samsung Electronics)의 지원을 통하여 수행되었음 (IO201210-07929-01).
† 연락저자 : 백준걸 교수, 02841 서울특별시 성북구 안암로 145 고려대학교 산업경영공학과, Tel : 02-3290-3396, Fax : 02-3290-4550, E-mail : jungeol@korea.ac.kr
2023년 1월 16일 접수; 2023년 3월 8일 수정본 접수; 2023년 3월 20일 게재 확정.

을 반영하는 것은 모델의 복잡도가 증가하기 때문에 실질적으로 적용하기에 어려움이 있다. 또한, 최근 두 가지 특성을 모두 반영하기 위해 2단계로 구성된 접근법이 많이 연구되고 있으나, 두 모델을 개별적으로 훈련하여 국소 최적에 빠지는 문제가 발생하기도 한다(Zhang *et al.*, 2021). 둘째, 제조 공정 데이터는 클래스 불균형(class imbalance) 문제가 자주 발생한다. 공정에서 수집되는 데이터는 이상 현상의 빈도가 낮고, 이상 관측치에 대한 라벨링이 제대로 되어있지 않은 경우가 대부분이기 때문이다.

이러한 문제들을 해결하기 위해 표현 학습(representation learning) 기반 비지도(unsupervised) 이상 탐지 방법론이 많이 연구되었다. 대표적으로 Generative Adversarial Networks(GAN) (Goodfellow *et al.*, 2014), 그리고 Variational Auto Encoder(VAE) (Kingma and Welling, 2013)를 이용한 모델들이 높은 차원을 갖는 데이터의 분포를 모델링하기 위해 사용되었다. 그러나 GAN 기반 모델은 데이터를 직접 생성하는 확률적인(stochastic) 절차만 정의하고, VAE는 잠재 벡터(latent vector) z에 대한 확률 분포(probability distribution)를 다루기 쉬운 확률 분포에 대략적으로만 근사한다는 한계가 존재한다.

Normalizing Flow(NF)는 연속된 가역적인(invertible) 변환으로 구성되어 데이터의 분포를 명시적으로(explicitly) 학습하므로 가능도(likelihood)를 정확하게 계산할 수 있다는 장점이 존재한다. NF는 정상 데이터에 대한 log likelihood를 최대화하는 훈련 프로세스를 통해, feature를 표준 정규 분포의 형태로 잠재 공간(latent space) 상에 임베딩(embedding)한다(Yu *et al.*, 2021). 정상 데이터만으로 NF를 학습한 모델은 검증 과정에서 정상 데이터가 입력되었을 때 큰 가능도가 반환될 것이다. 반면, 검증 시에 비정상 데이터가 입력되었을 경우 상대적으로 작은 가능도가 반환될 것이다. 따라서, NF는 가능도의 크고 작음을 기준으로 이상을 탐지하는 확률론적 이상 탐지에 적합하다.

Schmidt and Simic(2019)은 NF가 제조 도메인에서 시계열 이상 탐지에 적용될 수 있음을 보여주었다. 그러나, 시계열의 시간 의존성을 고려하지 않았다. 또한, Su *et al.*(2019)은 다변량 시계열 데이터에 NF를 적용한 OmniAnomaly를 통해 우수한 이상 탐지 성능을 보여주었다. 하지만, 모델이 복잡하기 때문에 실시간으로 이상 탐지를 수행해야 하는 제조 공정에는 적합하지 않다는 한계점이 존재한다. 따라서, 비교적 빠른 학습 시간만으로도 데이터의 특성을 반영할 수 있는 이상 탐지 방법이 필요하다.

본 논문은 제조 공정에 적용할 수 있는 NF 기반의 다변량 시계열 이상 탐지 방법론을 제안한다. Rasul *et al.*(2020)은 NF를 적용해 다변량 시계열 데이터에 대한 예측의 우수성을 입증했다. 또한, Masked Autoregressive Flow(MAF) (Papamakarios *et al.*, 2017)와 Real-Valued Non-Volume Preserving(RealNVP) (Dinh *et al.*, 2016)가 시간에 따른 변수 간의 관계를 잘 포착한다는 것을 증명했기에 이를 다변량 시계열 데이터에 대한 이상 탐지에 적용해 보고자 한다. 즉, 본 연구에서는 RealNVP와 MAF를 사용해 변수 간의 관계를 반영하고 데이터의 분포를 학습한다. 또한, 시계열의 시간 의존성을 반영하기 위해 Long Short Term Memory(LSTM)(Hochreiter and Schmidhuber, 1997), Bidirectional LSTM(Graves and Schmidhuber, 2005), 그리고 Input Attention Encoder(Qin *et al.*, 2017)를 사용하여 장기 의존성을 반영한 hidden state를 추출하였다. 추출된 LSTM의 hidden state와 입력 데이터를 함께 NF에 학습시킴으로써 이상을 탐지한다. 이러한 구조의 제안 방법은 선행연구 대비 모델 복잡도를 줄이며 변수 및 시간 상의 상관관계를 반영한다.

요약하자면, 본 논문의 기여는 다음과 같다.

- Flow 모델과 여러 LSTM 모델을 결합하여 변수 간 및 시간 상의 관계를 반영하고, 입력 데이터를 가우시안 분포로 매핑한다. 변환된 분포의 가능도 계산을 통해 이상을 탐지할 수 있다.
- 선행 연구 대비 모델이 단순화되어 제조 공정에서의 이상 탐지에 적용될 수 있다.
- 제조 공정에서 발생하는 데이터와 유사한 데이터셋에 대해 높은 이상 탐지 성능을 보였다.

본 연구의 구성은 다음과 같다. 먼저, 제2장에서는 NF와 LSTM 모듈, 그리고 이상 탐지 관련 선행 연구를 다룬다. 제3장에서는 제안 방법론인 MADFlow에 대해 설명한 뒤, 제4장에서 제안 방법론의 성능을 평가한다. 제5장에서는 결론과 추후 연구 방향을 제시한다.

2. 배경 지식 및 선행 연구

2.1 Normalizing Flow

NF는 Deep Generative Model 계열로, <Figure 1>과 같이 수행된다(Rasul *et al.*, 2020).

NF는 입력 데이터 ($\mathbf{x}$)가 역함수가 존재하는 flow layer ($f(\mathbf{x})$)를 연속적으로 거쳐 잠재 공간에서의 기본 분포(underlying distribution) $\mathbf{z}$를 학습하고 역변환(invertible)된 flow layer ($(f^{-1}(\mathbf{z}))$)를 통해 데이터를 복원(x') 할 수 있다. 주요한 특징은 입력(input) $\mathbf{x}$와 flow layer를 거친 출력(output) $\mathbf{z}$의 차원(dimension)이 동일하다는 것으로, 입력 공간(input space) $X \in \mathbb{R}^D$의 밀도(density) p_X가 간단한 분포를 따르는 잠재 공간 $Z \in \mathbb{R}^D$의 밀도 p_Z(예: isotropic Gaussian)로 변환되도록 $\mathbb{R}^D$에서 $\mathbb{R}^D$로 매핑(mapping) 된다. 이러한 매핑 $f : X \mapsto Z$은 일련

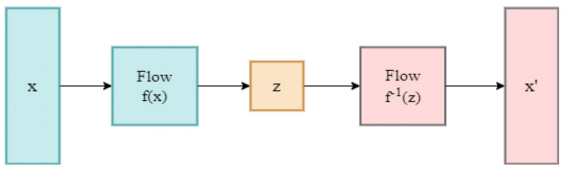


Figure 1. Structure of Normalizing Flow

의 전단사(bijection) 또는 역변환이 가능한 함수로 구성된다. 이를 변수 변환(change of variables)이라고 하는데, 변수 변환 공식에 기반하여 $\mathbf{x}=f^{-1}(\mathbf{z})$로 표현된다. 변수 변환된 $\mathbf{x}$의 밀도 $p_X(\mathbf{x})$를 식 (1)과 같이 나타낼 수 있다.

$$p_X(\mathbf{x}) = p_Z(\mathbf{z})\left|\det\left(\frac{\partial \mathrm{f}(\mathbf{x})}{\partial \mathbf{x}}\right)\right| \tag{1}$$

식 (1)에서 $\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}$는 대한 f의 야코비안 (Jacobian)을 의미한다. 야코비안 행렬(Jacobian matrix)은 벡터 $f(\mathbf{x})$, $\mathbf{x}$에 대한 일차 편미분 행렬이며, $\mathbf{x}$와 $\mathbf{z}$의 확률 밀도 함수(probability density function)의 관계가 야코비안 행렬식 (Jacobian determinant) 만큼의 비율을 갖는다. 각 layer마다 야코비안 행렬식을 계산하는 것은 많은 비용이 들기 때문에, NF는 다음과 같은 두 가지 조건을 만족하여야 한다. 첫째, f는 가역적이어야 한다. 둘째, f에 대한 야코비안 행렬식 계산이 쉬워야 한다. 위의 두 조건을 만족하지 않으면 NF의 학습이 제대로 이루어지지 않는다.

RealNVP는 야코비안 행렬식의 연산을 간단히 하기 위해 affine coupling layer라는 flow network 구성 방법을 제안했다 (Dinh *et al.*, 2016). 이는 입력의 차원을 두 개 부분으로 나누어서 입력의 일부는 변환하지 않고, 변환되지 않은 변수들의 함수를 통해 나머지 변수를 변환한다. 이를 식 (2)와 같이 나타낼 수 있다.

$$\begin{cases}\mathbf{y}^{1:\mathrm{d}} = \mathrm{b}\odot\mathbf{x}^{1:\mathrm{d}} \\ \mathbf{y}^{\mathbf{d+1:D}} = (1-\mathrm{b})\left[(\mathbf{x}^{\mathrm{d}+1:\mathrm{D}}\odot\exp\{s(b\odot x^{1:d})\}+t(b\odot\mathbf{x}^{1:\mathrm{d}}))\right]\end{cases} \tag{2}$$

식 (2)에서 $\mathbf{x}^{1:d}$는 변환되지 않은 변수, $\mathbf{x}^{d+1:D}$는 변환되는 변수를 의미한다. 또한, $\odot$는 성분 별 곱(element wise product)이고, $s(\bullet)$는 scaling, $t(\bullet)$는 translation function으로 각각 linear하지 않은 복잡한 함수이다. b는 binary masking으로 역전파 시에 예측되어야 할 mask된 부분을 의미하며 식 (3)을 통해 업데이트 된다.

$$\mathbf{x}_{masked} = (x_{masked} + t(\mathbf{x}^{1:d}))\odot\exp(s(\mathbf{x}^{1:d})) \tag{3}$$

Coupling layer에서는 masked input에 대한 affine transform을 수행하기 위해 $s(\bullet)$, $t(\bullet)$함수를 학습한다.

Linear하지 않은 function f를 모델링하기 위해서 $X\mapsto Y_1\mapsto Y_2\mapsto\cdots\mapsto Y_{K-1}\mapsto Z$가 되도록 다수의 coupling layer가 구성된다. 또한, 변경되지 않는 차원(e.g. $\mathbf{x}^{1:d}$)이 항상 동일하지 않도록 셔플링(shuffling)이 수행된다. 변수 변환 공식 (change of variable formula)에 입각해 데이터 포인트 $\mathbf{x}$에 대한 확률 밀도 함수를 식 (4)와 같이 나타낼 수 있다.

$$\log p_X(\mathbf{x}) = \log p_Z(\mathbf{z}) + \log\left|\det\left(\frac{\partial z}{\partial x}\right)\right| \tag{4}$$

$$= \log p_Z(\mathbf{z}) + \sum_{i=1}^{K}\log|\det(\partial\,\mathbf{y}_i/\partial\,\mathbf{y}_{i-1})|$$

Affine coupling layer에 의해 RealNVP의 야코비안은 식 (5)와 같은 하삼각행렬(lower triangular matrix) 형태로 변환이 되고, 야코비안 연산은 식 (6)과 같이 scaling function output의 합으로 매우 간단해진다.

$$\frac{\partial\,\mathbf{y}_i}{\partial\,\mathbf{y}_{i-1}} = \begin{bmatrix} I & 0 \\ \dfrac{\partial\, y_i^{d+1:D}}{\partial\, y_{i-1}^{1:d}} & diag(\exp(s(\mathbf{y}_{i-1}^{1:d}))) \end{bmatrix} \tag{5}$$

$$\log|\det(\partial\, y_i/\partial\, y_{i-1})| = \log|\exp(\sum(s(\mathbf{y}_{i-1}^{1:d}))| \tag{6}$$

NF 모델의 훈련은 일반적으로 훈련 데이터의 가능도를 최대화하는 방향으로 학습된다. 즉, T 기간 동안의 데이터에 대한 negative log likelihood를 최소화하여 학습하게 된다. 최종적으로 NF의 손실 함수(loss function)는 식 (7)과 같다.

$$L = \frac{1}{T}\sum_{t=1}^{T} -\log p_X(\mathbf{x};\theta) \tag{7}$$

2.2 Long Short Term Memory

(1) Vanilla LSTM

LSTM은 Recurrent Neural Network (RNN)에서의 기울기 소실(vanishing gradient) 문제를 해결하기 위해 고안된 모델이다. LSTM의 구조는 <Figure 2>와 같다.

매 타임 스텝(time step)마다 2개의 state: cell state($\mathbf{C}_t$), hidden state($\mathbf{h}_t$)를 유지하며 3개의 gate: forget gate($\mathbf{f}$), input gate($\mathbf{i}$), output gate($\mathbf{o}$)을 통해 계산이 이루어진다. t시점의 입력 벡터가 $\mathbf{x}_t\in\mathbb{R}^n$일 때, hidden state $\mathbf{h}_t\in\mathbb{R}^m$는 식 (8)~(12)를 통해 도출된다.

$$\mathbf{i}_t = sigm(\mathbf{W}_i[\mathbf{h}_{t-1};\mathbf{x}_t]+\mathbf{b}_i) \tag{8}$$

$$\mathbf{f}_t = sigm(\mathbf{W}_f[\mathbf{h}_{t-1};\mathbf{x}_t]+\mathbf{b}_f) \tag{9}$$

$$\mathbf{o}_t = sigm(\mathbf{W}_o[\mathbf{h}_{t-1};\mathbf{x}_t]+\mathbf{b}_o) \tag{10}$$

$$\mathbf{C}_t = \mathbf{f}_t\odot C_{t-1} + \mathbf{i}_t\odot\tanh(\mathbf{W}_C[\mathbf{h}_{t-1};\mathbf{x}_t]+\mathbf{b}_C) \tag{11}$$

$$\mathbf{h}_t = \boldsymbol{o}_t \circ \tanh(\mathbf{C}_t) \tag{12}$$

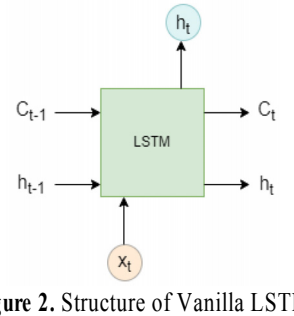


Figure 2. Structure of Vanilla LSTM

식 (8)~(11)의 $[\mathbf{h}_{t-1};\mathbf{x}_t] \in \mathbb{R}^{m+n}$은 이전 시점의 hidden state인 $\mathbf{h}_{t-1}$과 현재 입력 $\mathbf{x}_t$를 연결(concatenate) 해준 값이며 $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o$, $\mathbf{W}_C \in \mathbb{R}^{m\times(m+n)}$과 $\mathbf{b}_i$, $\mathbf{b}_f$, $\mathbf{b}_o$, $\mathbf{b}_C \in R^m$ 는 학습 파라미터를 의미한다. 식 (8)에서 input gate는 셀 상태를 업데이트하는데 사용된다. t시점의 입력과 $t-1$시점의 hidden state가 sigmoid함수로 전달되어 0과 1 사이의 값으로 변환하고 새로 업데이트 될 값을 결정한다. 식 (9)의 forget gate는 0에서 1 사이의 숫자를 제공한다. 1은 해당 정보를 모두 유지하는 것을 의미하며, 0은 모두 잊는 것을 의미한다. 식 (10)의 output gate는 이전 입력에 대한 정보를 갖고 있고 다음 hidden state를 결정한다. 식 (11)의 $\tanh(\mathbf{W}_C[\mathbf{h}_{t-1};\mathbf{x}_t]+\mathbf{b}_C)$ 는 input modulation으로 -1에서 1 사이의 값을 갖으며, 현재 정보를 cell state에 얼마나 더할 것인지를 결정하는 역할을 수행한다. 최종적으로 t시점의 hidden sate는 식 (12)와 같이 계산된다.

(2) Bidirectional LSTM

Bidirectional LSTM은 양방향성을 갖는 LSTM으로 <Figure 3>과 같은 구조로 이루어진다.

<Figure 3>에서의 파란색 화살표는 forward LSTM을 의미하며, 초록색 화살표는 backward LSTM을 의미한다. 이후 각각 forward 방향의 t번째 cell과 backward 방향의 $N-t$번째 cell을 연결하여 최종 출력을 얻게 된다. Bidirectional LSTM은 시퀀스 (sequence) 데이터에 대한 정보를 양방향으로 추출할 수 있어 일반적으로 Vanilla LSTM보다 성능이 우수하다는 특징이 있다.

(3) Input Attention Encoder

Qin *et al.*(2017)은 서로 관련이 있는 입력 변수(input feature)를 적응적으로(adaptively) 선택할 수 있는 Input Attention based Encoder를 제안하였다. Input Attention Encoder는 <Figure 4>와 같이 구성된다.

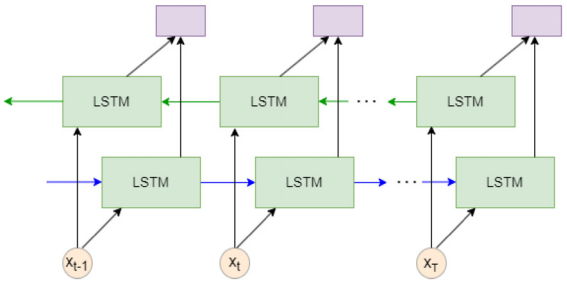


Figure 3. Structure of Bidirectional LSTM

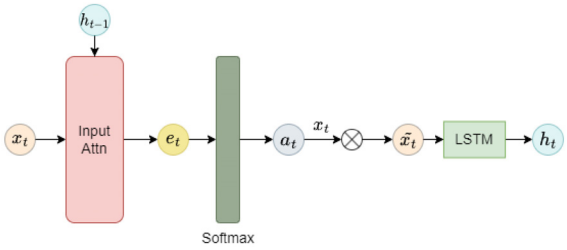


Figure 4. Structure of Input Attention Encoder

k번째 차원의 $\mathbf{x}^k = (x_1^k, x_2^k, \cdots, x_T^k)^\top \in \mathbb{R}^T$가 주어졌을 때, 식 (13), (14)와 같이 Encoder LSTM 유닛의 $\mathbf{h}_{t-1}$과 $\mathbf{C}_{t-1}$를 참조하고, multilayer perceptron을 통해서 attention mechanism을 구성할 수 있다.

$$e_t^k = \mathbf{v}_e^\top \tanh(\mathbf{W}_e[\mathbf{h}_{t-1};\mathbf{C}_{t-1}]+\mathbf{U}_e\mathbf{x}^k) \tag{13}$$

$$a_t^k = \frac{\exp(e_t^k)}{\sum_{i=1}^n \exp(e_t^i)} \tag{14}$$

식 (13)에서 $\mathbf{v}_e \in \mathbb{R}^T$, $\mathbf{W}_e \in \mathbb{R}^{T\times 2m}$, 그리고 $\mathbf{U}_e \in \mathbb{R}^{T\times T}$는 각각 학습 파라미터이다. 식 (14)에서 a_t^k는 t시점 k에서의 번째 입력 변수 중요도를 계산하는 attention weight로, 합이 1이 될 수 있도록 e_t^k에 softmax 함수가 적용된다. 이러한 attention weight를 사용해 식 (15)에서 변수 중요도가 반영된 입력 값을 반환할 수 있고, 식 (16)을 통해 t시점의 hidden state가 업데이트 된다.

$$\tilde{\mathbf{x}}_t = (a_t^1 x_t^1, a_t^2 x_t^2, \cdots, a_t^n x_t^n) \tag{15}$$

$$\mathbf{h}_t = f_1(\mathbf{h}_{t-1}, \tilde{\mathbf{x}}_t) \tag{16}$$

식 (16)에서의 f_1은 식 (8)~(12)에 따라 계산된 LSTM 유닛이다.

2.3 이상치 탐지 관련 연구

제조 공정은 레이블(label)이 지정된 이상 데이터가 본질적으로 부족하다. 따라서 대부분의 시계열 이상 탐지 방법론은 비지도 학습 방식으로 이루어진다.

또한, 데이터의 내재된 특성을 잘 반영할 수 있도록 하는 표현 학습 기반 방법론이 최근에 많이 연구되고 있으며 높은 성능을 보여주었다.

Li *et al.*(2019)은 LSTM based GAN을 이용하여 다변량 시계열에 대한 이상 탐지를 수행하였다. 생성자(Generator)와 판별자(Discriminator)가 모두 2개의 LSTM layer로 구성되며, 생성자의 손실은 최소화하고 판별자의 손실은 최대화하는 경쟁 학습(adversarial training)을 통해 각각 학습된다. Guo *et al.*(2018)은 GRU를 기반으로 한 Gaussian Mixture VAE를 제안하였다. GRU 셀은 시간 시퀀스 간의 상관관계를 반영하기 위해 사용되었고, 잠재 공간에서 Gaussian Mixture priors를 사용해 멀티 모달 데이터를 특성화 했다. 해당 구조를 통해 재구성 오차가 특정 임계값(threshold) 이상일 때, 이상이라 분류했다. Malhotra *et al.*(2016)은 LSTM based Autoencoder를 사용해 다변량 시계열 데이터를 비지도 방식으로 학습하고 이상치를 탐지하였다. 제안 방법은 LSTM 인코더(Encoder)와 LSTM 디코더(Decoder)로 구성된다. 인코더는 다변량 데이터를 저차원 표현으로 변환하고, 디코더는 인코더를 거쳐 얻은 표현을 이용해 다변량 데이터

를 재구성한다. 인코더의 입력과 디코더에서 나온 재구성된 입력의 차이를 줄이도록 학습한다. Zong *et al.*(2018)은 Deep Autoencoder를 통해 차원을 축소한 후, Gaussian Mixture Model (GMM)로 저차원 공간의 데이터 밀도를 추정하는 방법을 제안하였다. 가우시안 분포를 기반으로 데이터의 분포를 추정한다는 점에서 NF와 유사하나, GMM은 여러 정규분포를 가정하고 정규분포의 개수를 사용자가 설정해야 한다는 점에서 NF와 차이가 있다.

3. 제안 방법

본 연구는 변수 간의 관계와 시간 의존성을 학습할 수 있는 NF 기반의 다변량 시계열 이상 탐지 방법론인 MADFlow을 제안한다. 제안 방법의 전체 구조는 <Figure 5>와 같다. MADFlow는 다음과 같은 3단계로 구성된다. (1) Temporal Encoder: 데이터가 LSTM을 거쳐 시계열을 요약하는 정보를 담고 있는 hidden state를 반환한다. (2) Flow Module: 도출된 hidden state를 조건부로 하는 데이터의 분포를 NF을 통해 모델링한다. (3) Training: NF의 출력(output)으로 도출된 분포에 대한 negative log likelihood를 손실 함수로써 학습한다. 각 방법은 유기적인 순서로 학습이 진행된다.

3.1 Temporal Encoder

본 논문에서는 시간 의존성을 반영하기 위해 Temporal Encoder 구조를 사용한다. Temporal Encoder에서는 LSTM 기반의 모델들을 사용한다. 본 연구에서는 hidden state를 도출하기 위해 <Figure 2>의 Vanilla LSTM, <Figure 3>의 Bidirectional LSTM, 그리고 <Figure 4>의 Input Attention Encoder를 사용한다. Vanilla LSTM은 가장 기본적인 LSTM의 형태로, 단기 상태를 고려하는 동시에 장기 상태에 대한 정보를 공급받을 수 있는 모델이다. Bidirectional LSTM은 Vanilla LSTM을 양방향으로 겹쳐 놓은 구조로 순차적으로 입력된 데이터에 대해서 이전 데이터와의 관계뿐만 아니라 이후 데이터와의 관계까지 학습한다. Input Attention은 변수에 대해 attention을 주어 중요한 변수를 중점적으로 학습할 수 있다.

즉, 제안 방법은 시간 의존성뿐만 아니라 공간적(spatial)인 정보까지 추가로 고려하여 hidden state를 추출할 수 있는 여러 방법을 고려한다. Vanilla LSTM과 Bidirectional LSTM은 시간 상의 정보를 중점적으로 반영하는 반면, Input Attention Encoder는 시간 상의 정보를 반영하는 동시에 중요한 변수를 중점적으로 학습할 수 있다. 변수 시점 t에서 D개의 차원을 갖는 데이터 $\mathbf{x}_t \in \mathbb{R}^D$로부터 t시점의 hidden state $\mathbf{h}_t$는 식 (17)과 같이 계산된다.

$$\mathbf{h}_t = LSTM(\mathbf{x}_t, \mathbf{h}_{t-1}) \tag{17}$$

식 (17)에서 $\mathbf{h}_t$는 t시점까지의 시계열을 요약하는 정보를 담고 있으며, <Figure 5>과 같이 t시점의 데이터 $\mathbf{x}_t$와 연결되어 Flow Module의 coupling layer에 입력된다.

3.2 Flow Module

데이터의 분포를 학습하기 위해, Flow Module을 사용해 역함수가 존재하는 함수를 학습한다. 제안 방법은 Temporal Encoder에 사용된 LSTM과 함께 <Figure 5>에서 볼 수 있듯, K개의 Flow Module(RealNVP 혹은 MAF)을 결합하여 분포를 모델링한다. LSTM으로부터 나온 hidden state를 입력 데이터 $\mathbf{x}$와 연결한 후, Flow Module의 coupling layer에 입력한다. Flow Module의 coupling layer에서는 파라미터의 공유를 통해 변수 간의 관계를 파악할 수 있다. Coupling layer는 식 (2)와 같이 scaling과 translation function으로 구성되어 있으며, 식 (18)과 같이 scaling과 translation function $s(\bullet)$, $t(\bullet)$에 입력 데이터 $\mathbf{x}^{1:d}$과 hidden state $\mathbf{h}$가 결합된 채 입력된다.

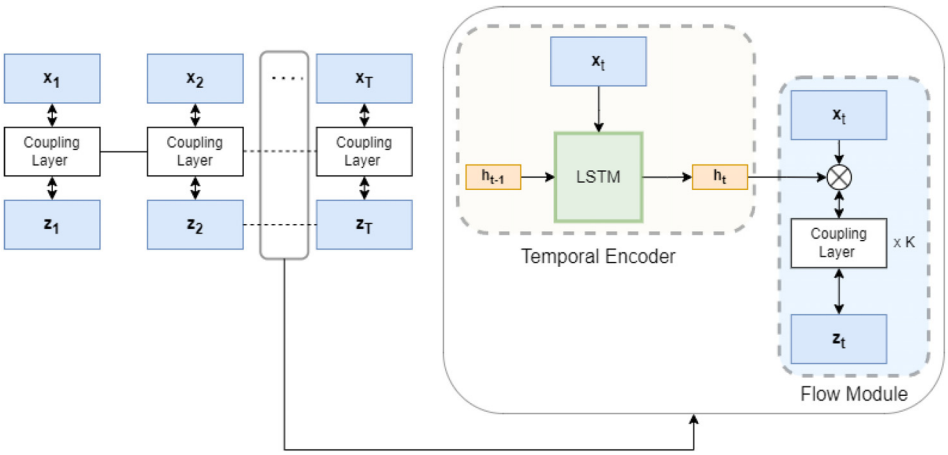


Figure 5. Overview of Multivariate Time Series Anomaly Detection via Normalizing Flow

$$s(\mathrm{concat}(\mathbf{x}^{1:d},\ \mathbf{h})) \quad (18)$$
$$t(\mathrm{concat}(\mathbf{x}^{1:d},\ \mathbf{h}))$$

RealNVP에서는 affine coupling layer가, MAF에서는 Masked Masked Autoencoder for Distribution Estimation (MADE) (Germain *et al.*, 2015)가 coupling layer로서 야코비안을 하삼각 행렬 형태로 구성하고, 복잡한 연산을 간단히 할 수 있다. 최종적으로 t 시점에서의 log density는 식 (19)와 같다.

$$\log p_X(\mathbf{x}_t \mid \mathbf{h}_t) = \log p_Z(f(\mathbf{x}_t \mid \mathbf{h}_t) + \log |det(\frac{\partial f(\mathbf{x}_t|\mathbf{h}_t)}{\partial \mathbf{x}_t})| \quad (19)$$

3.3 Training

제안 방법은 3.1장과 3.2장을 거쳐 유기적으로 학습되며, 최종 손실 함수는 식 (7)에서 식 (20)과 같이 변경된다.

$$L = \frac{1}{T}\sum_{t=1}^{T} -\log p_X(\mathbf{x}_t \mid \mathbf{h}_t) \quad (20)$$

모델은 데이터 $\mathbf{x}$에 대한 가능도를 최대화한다. 손실을 최소화하는 관점에서 가능도에 음수를 취하여, negative log likelihood를 손실 함수로써 학습한다. 또한, T 길이의 윈도우(window)내의 모든 데이터에 대해서 연산해야 하기 때문에 시점의 길이인 T로 나누어 손실의 평균을 계산한다. 이를 기준으로 제안 방법의 학습을 진행하였다.

4. 실험

4.1 데이터 및 평가 지표

제안하는 모델을 평가하기 위해 Server Machine Dataset (SMD)을 검증에 사용하였다. SMD는 인터넷 기업에서 5주 동안 38개의 metric을 사용해서 기계의 고장 여부를 수집한 데이터셋이다. 측정을 위해 사용된 지표로는 CPU load, network usage, memory usage 등이 있다. 해당 데이터의 이상 비율은 4.16%로 클래스 불균형 문제가 존재한다. 또한, 순간적으로 값이 떨어지거나 튀어 오르는 시점을 이상으로 판단하는 특성을 갖고 있다.

Table 1. Confusion Matrix

	Actual True	Actual False
Predicted True	TP (True Positive)	FP (False Positive)
Predicted False	FN (False Negative)	TN (True Negative)

본 연구는 성능 평가 지표로써 Area Under Receiver Operating Characteristic Curve(AUROC)를 사용한다. AUC는 가능한 모든 임계값에 대해 산출되기 때문에 임계값에 독립적인 성능 평가 지표로 활용될 수 있다. AUC는 ROC curve의 밑면적으로, 값이 1에 가까울수록 좋은 성능의 모델이라고 할 수 있다. ROC curve는 모델의 임계값을 연속적으로 바꾸면서 측정한 False Positive Rate (FPR)과 True Positive Rate(TPR)의 변화를 나타낸 것으로, (0,0)과 (1,1)를 연결하는 곡선 형태로 나타난다. FPR과 TPR은 각각 <Table 1>의 혼동 행렬(Confusion Matrix) 결과에 따라 계산된다.

TPR은 민감도(sensitivity)로 <Table 1>에서 실제로 참인 케이스를 참으로 잘 예측할 비율을 의미하며, FPR은 1-특이도(specificity)로 <Table 1>에서 실제로 참이 아닌 케이스에 대해 참으로 잘못 예측한 비율을 의미한다.

4.3 실험 결과

제안 방법을 평가하기 위한 실험은 3단계로 구성된다. 1단계에서는 MAF, RealNVP 각각에 대해서 배치 크기(batch size)를 변경해가며 10번씩 반복 실험했을 때의 학습 오차(training loss)와 평균 AUC를 계산하는 실험을 진행했다. 실험을 통해 RealNVP가 MAF에 비해 학습오차가 빠르고 안정적으로 수렴함을 확인할 수 있다. 2단계에서는 Temporal Encoder와 Flow Module의 조합으로 나올 수 있는 총 6가지 구조 중 최적의 구조를 찾기 위한 실험을 진행하였다. 마지막으로 3단계에서는 다른 모델들과 제안하는 방법론의 성능을 비교하는 실험을 진행했다.

실험에 사용된 GPU 사양은 ASUS TUF GeForce RTX 3080 Ti이다. MADFlow는 학습을 위해 6개의 Flow Module을 사용하였으며, Temporal Encoder와 Flow Module에 공통적으로 적용되는 hidden dimension은 52로 설정했다. 초기 learning rate는 0.001로 설정했고 여러 optimizer를 사용해 성능을 평가한 결과, Adam을 사용했을 때 성능이 가장 뛰어났다. 실험에 사용된 윈도우 크기는 30으로 설정했다.

(1) 배치 크기 변경에 따른 비교 실험

배치 크기를 변경하며 실험했을 때의 성능과 학습 오차를 분석하기 위해, Temporal Encoder로는 Bidirectional LSTM을, Flow Module로는 MAF와 RealNVP를 사용하였다. 반복 횟수(epoch)는 각각 MAF는 200회, RealNVP는 20회로 설정했다. 총 4개의 배치 크기(32, 64, 128, 256)에 대해서 MAF와 RealNVP 모두 10번씩 반복 실험을 진행한 결과, 평균 AUC는 <Table 2>와 같다.

실험 결과, MAF는 배치 크기가 32일 때 가장 성능이 우수하였고, RealNVP의 경우에는 배치 크기가 128일 때 평균 AUC가 0.9250으로 성능이 가장 우수하였다.

또한, 배치 크기 별로 학습 오차의 분포를 상자 그림(boxplot)으로 시각화한 결과는 <Figure 6>과 같다.

Table 2. Mean AUC per Batch Size

Flow Module	Batch Size	AUC
MAF	32	**0.7264 (0.0860)**
	64	0.7127 (0.1138)
	128	0.6813 (0.1055)
	256	0.6514 (0.0575)
RealNVP	32	0.7960 (0.0964)
	64	0.8233 (0.0724)
	128	**0.9250** (0.0360)
	256	0.8889 (0.0741)

<Figure 6>의 (a) 상자 그림은 Flow Module로 MAF를 사용했을 때의 배치 크기 별 학습 오차를 보여준다. MAF에 대한 학습 반복 횟수를 200으로 설정한 이유는 초반 학습이 불안정 하여 RealNVP와 동일하게 반복을 20으로 설정했을 때 학습이 잘 되지 않기 때문이다. <Figure 6>의 (b) 상자 그림은 Flow Module로는 RealNVP를 사용했을 때의 배치 크기 별 학습 오차를 보여준다.

결과적으로 상자 그림 영역의 넓이로 알 수 있듯이, <Figure 6> (b)에서 RealNVP의 반복 수가 <Figure 6> (a)의 MAF에 비해 적음에도 빠르게 수렴하여 loss가 낮은 값에서 안정적으로 분포함을 확인할 수 있다.

즉, MAF는 RealNVP와는 다르게 학습이 불안정하기 때문에 배치 크기가 작을 때 (e.g. 32) 학습 안정도와 일반화 성능이 증가해 우수한 성능을 보였다(Masters and Luschi, 2018).

(2) 최적 제안 구조 탐색을 위한 비교 실험

Temporal Encoder와 Flow Module의 조합으로 나올 수 있는 6가지 구조에 대한 실험 결과는 <Table 3>와 같다.

Table 3. Optimal Structure of Proposed Method

Method	AUC
MAF + Vanilla LSTM	0.6656 (0.0629)
MAF + bidirectional LSTM	0.7264 (0.0086)
MAF + Input Attention	0.6678 (0.0910)
RealNVP + Vanilla LSTM	0.8413 (0.0889)
RealNVP + bidirectional LSTM	**0.9250 (0.0360)**
RealNVP + Input Attention	0.8545 (0.0991)

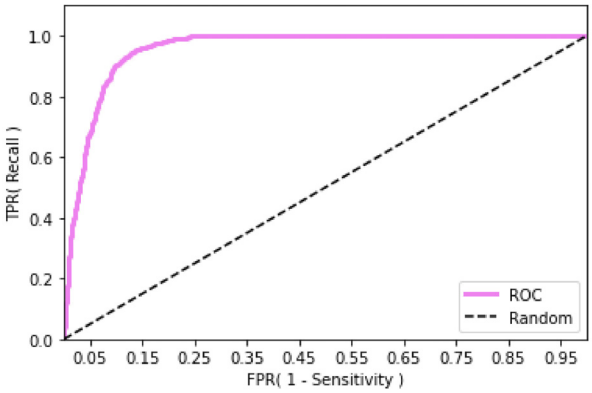
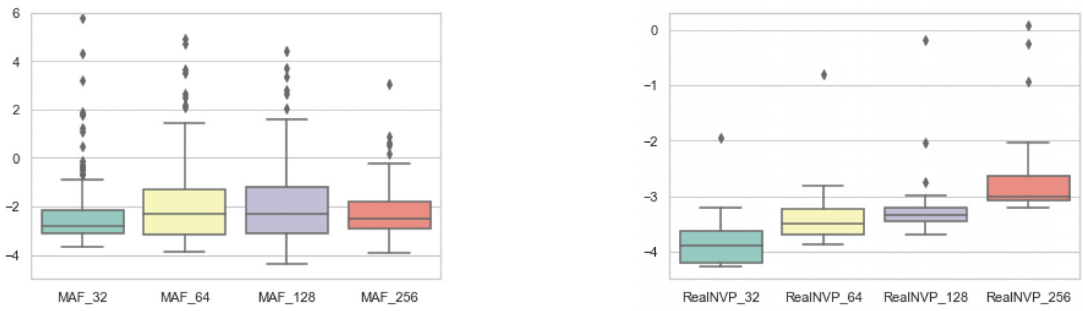


Figure 7. ROC Curve of RealNVP + bidirectional LSTM

MAF, RealNVP에 대한 배치 크기는 <Table 2>에서 각각 성능이 가장 우수했던 32와 128로 설정했고 MAF 기반 모델은 200 반복, RealNVP 기반 모델은 20 반복을 설정해 성능을 비교하였다. 실험 결과를 통해 알 수 있듯이 RealNVP와 Bidirectional LSTM을 사용했을 때 평균 AUC가 0.9250으로 성능이 가장 우수했으며 이때의 ROC 커브는 <Figure 7>과 같다.

이는 양방향으로 시간 상의 관계를 학습하는 Bidirectional LSTM의 특성이 데이터의 시간 의존성을 성공적으로 반영했



(a) (b)

Figure 6. Boxplot of Train Loss (a) MAF, (b) RealNVP

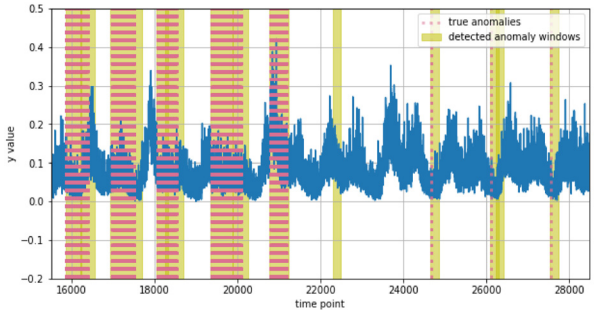


Figure 8. Anomaly Detection Results for SMD

음을 증명한다. 또한 RealNVP에 Vanilla LSTM 혹은 Input Attention Encoder를 사용했을 때에도 성능이 각각 0.8413, 0.8545로 MAF를 사용한 구조에 비하여 우수한 성능을 보였다. 즉, RealNVP를 사용할 경우 학습 시간 (반복 수) 대비 성능을 뛰어남을 확인했다.

성능이 가장 우수한 MADFlow 구조인 RealNVP와 Bidirectional LSTM의 결합으로 테스트 데이터에 대한 이상을 탐지한 결과는 <Figure 8>과 같다.

<Figure 8>에서 한 가지 케이스를 제외하고는 탐지한 이상 윈도우(anomaly window)가 실제 이상 시점을 대부분 정확하게 반영함을 확인했다.

(3) 다른 모델과의 성능 비교 실험

제안 모델 중 성능이 가장 뛰어났던 RealNVP + Bidirectional LSTM의 비교군으로 Multivariate anomaly detection for time series data with generative adversarial networks (MAD-GAN) (Li *et al.*, 2019), LSTM Autoencoder와 단독 RealNVP, 단독 MAF, Deep one-class classification(Deep-SVDD)(Ruff *et al.*, 2018), 그리고 Deep Autoencoding Gaussian Mixture Model(DAGMM) (Zong *et al.*, 2018)를 설정했다. LSTM Autoencoder와 MAD-GAN, Deep-SVDD, DAGMM과 같은 모델들은 다변량 시계열 데이터에 대한 비지도 기반 이상 탐지에 자주 사용되는 방법이기에 제안 모델과 성능을 비교하였다. 또한, 시간 의존성을 반영하지 않고 Temporal Encoder 없이 Flow Module (RealNVP, MAF)만을 사용했을 때의 성능과 제안 모델을 비교하였다. <Table 4>는 다른 모델들과 제안 방법의 성능과 학습 시간 및 파라미터 수를 비교한 표이다. 반복 수는 모두 동일하게 20으로 설정해주었다.

Table 4. Comparisons with Other Methods

Method	Parameter #	Training Time	AUC
LSTM Autoencoder	990,465	4558s	0.8587 (0.0087)
MAD-GAN	277,539	190s	0.7906 (0.0208)
Deep-SVDD	57952	7s	0.8787 (0.0262)
DAGMM	923	14s	0.8731 (0.0790)
RealNVP only	365,052	89s	0.6274 (0.0001)
MAF only	182,844	49s	0.6817 (0.0938)
MADFlow (ours)	387,932	136s	**0.9250 (0.0360)**

<Table 4>에서 AUC를 기준으로 성능을 비교했을 때, 제안 방법론이 타 모델 대비 유의미하게 성능이 우수함을 확인할 수 있다. 또한 비교적 길지 않은 학습 시간으로도 높은 성능을 보임을 실험적으로 입증했다

5. 결론

본 연구는 변수 간의 상관관계와 시간 상의 의존관계를 반영하여 이상을 탐지할 수 있는 MADFlow를 제안하였다. MADFlow는 Temporal Encoder와 Flow Module을 결합한 구조로 Temporal Encoder에서는 LSTM 기반 모델을 사용했고 Flow Module에서는 MAF와 RealNVP를 적용했다.

실제 제조 공정과 같이 장비의 여러 센서들로부터 수집된 클래스 불균형 데이터에 대해서 최적 MADFlow 구조를 찾아냈고, LSTM Autoencoder, MAD-GAN, Deep-SVDD, DAGMM, 단일 MAF, 단일 RealNVP와 비교하여 성능의 우수성을 입증했다. 그러나 Deep-SVDD, DAGMM 모델과 비교하면 학습 시간이 길기 때문에 모델 학습이 선행적으로 완료된 상태에서 실제 공정에 적용할 시 정확도 높은 이상 탐지가 가능할 것으로 기대한다. 또한, 제조 도메인 외에도 유사한 데이터 특성을 가진 다양한 도메인의 다변량 센서 데이터에 대해 효과적으로 적용이 가능할 것으로 기대한다.

향후 연구에서는 MADFlow가 실시간 이상 탐지에 적용될 수 있도록 추가 연구가 필요하며, 다양한 데이터셋에 대한 성능 평가를 통해 제안 방법의 우수성을 확보하고자 한다.

참고문헌

Dinh, L., Sohl-Dickstein, J., and Bengio, S. (2016), Density estimation using real nvp arXiv preprint arXiv:1605.08803.

Germain, M., Gregor, K., Murray, I., and Larochelle, H. (2015), Made: Masked autoencoder for distribution estimation. In *International Conference on Machine Learning*, PMLR, 881-889.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... and Bengio, Y. (2014), Generative adversarial nets, *Advances in Neural Information Processing Systems*, 27.

Graves, A. and Schmidhuber, J. (2005), Framewise phoneme classification with bidirectional LSTM and other neural network architectures, *Neural Networks*, **18**(5-6), 602-610.

Guo, Y., Liao, W., Wang, Q., Yu, L., Ji, T., and Li, P. (2018), Multidimensional time series anomaly detection: A gru-based gaussian mixture variational autoencoder approach. In *Asian Conference on Machine Learning*, PMLR, 97-112.

Hochreiter, S. and Schmidhuber, J. (1997), Long short-term memory, *Neural Computation*, **9**(8), 1735-1780.

Kingma, D. P. and Welling, M. (2013), Auto-encoding variational bayes, arXiv preprint arXiv:1312.6114.

Li, D., Chen, D., Jin, B., Shi, L., Goh, J., and Ng, S. K. (2019), MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks, In *International Conference on Artificial Neural Networks*, Springer, Cham, 703-716.

Malhotra, P., Ramakrishnan, A., Anand, G., Vig, L., Agarwal, P., and Shroff, G. (2016), LSTM-based encoder-decoder for multi-sensor anomaly detection, arXiv preprint arXiv:1607.00148.

Masters, D. and Luschi, C. (2018). Revisiting small batch training for deep neural networks, arXiv preprint arXiv:1804.07612.

Papamakarios, G., Pavlakou, T., and Murray, I. (2017), Masked autoregressive flow for density estimation, *Advances in Neural Information Processing Systems*, 30.

Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., and Cottrell, G. (2017), A dual-stage attention-based recurrent neural network for time series prediction, arXiv preprint arXiv:1704.02971.

Rasul, K., Sheikh, A. S., Schuster, I., Bergmann, U., and Vollgraf, R. (2020), Multivariate probabilistic time series forecasting via conditioned normalizing flows, arXiv preprint arXiv:2002.06103.

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., ... and Müller, K. R. (2021), A unifying review of deep and shallow anomaly detection, *Proceedings of the IEEE*, **109**(5), 756-795.

Ruff, L., Vandermeulen, R., Goernitz, N., Deecke, L., Siddiqui, S. A., Binder, A., ... and Kloft, M. (2018), Deep one-class classification, In *International Conference on Machine Learning*, PMLR, 4393-4402.

Schmidt, M. and Simic, M. (2019), Normalizing flows for novelty detection in industrial time series data, arXiv preprint arXiv:1906.06904.

Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., and Pei, D. (2019), Robust anomaly detection for multivariate time series through stochastic recurrent neural network, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

Yu, J., Zheng, Y., Wang, X., Li, W., Wu, Y., Zhao, R., and Wu, L. (2021), Fastflow: Unsupervised anomaly detection and localization via 2d normalizing flows, arXiv preprint arXiv:2111.07677.

Zhang, Y., Chen, Y., Wang, J., and Pan, Z. (2021), Unsupervised deep anomaly detection for multi-sensor time-series signals, *IEEE Transactions on Knowledge and Data Engineering*.

Zong, B., Song, Q., Min, M. R., Cheng, W., Lumezanu, C., Cho, D., and Chen, H. (2018), Deep autoencoding gaussian mixture model for unsupervised anomaly detection, In *International conference on learning representations*.

저자소개

문지원 : 숭실대학교 산업정보시스템공학과에서 2022년 학사학위를 취득하고 고려대학교에서 산업경영공학과 석사과정에 재학 중이다. 연구분야는 시계열 분석, 이상치 탐지이다.

송승환 : 고려대학교 통계학과에서 2019년 학사학위를 취득하고 고려대학교에서 산업경영공학과 석박사통합과정에 재학 중이다. 연구분야는 생성모델, 시계열 분석이다.

백준걸 : 고려대학교 산업경영공학부에서 1993년 학사, 1995년 석사, 2001년 박사학위를 취득하였다. 인덕대학교와 광운대학교에서 교수를 역임하였으며, 2008년부터는 고려대학교 산업경영공학부 교수로 재직하고 있다. 연구 분야는 Intelligent Diagnosis and Prognosis, Data Science for Manufacturing이다.