

시계열 데이터(time-series data)

What is time series?

관측치가 시간적 순서를 가진 데이터. 순차적(sequentially)으로 관측한 값들의 집합. 시간을 독립변수로 사용한다.

How Use?

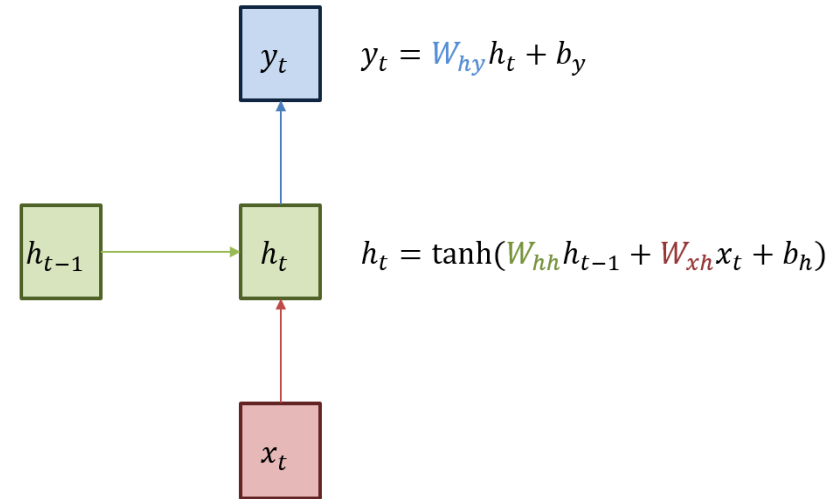
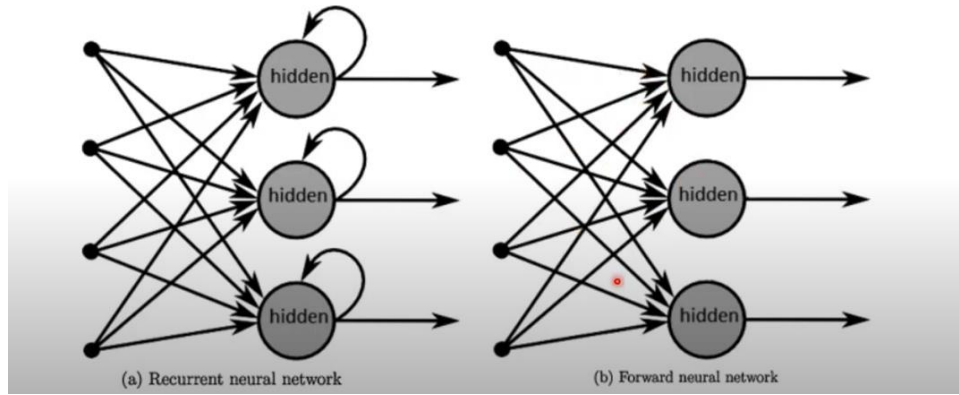
시계열 데이터는 과거의 데이터를 통해서 현재의 움직임, 그리고 미래를 예측하는 데 사용된다. 일반적인 label 데이터는 input과 label간의 상관관계를 다루는 반면에 시간에 따라 움직이는 과거의 자료를 가지고 예측하게 된다.

시계열 데이터(time-series data)

시계열 분석의 목적 : 순차 데이터의 과거에서 의존성의 패턴을 알아내는 것과, 그 패턴을 이용하여 미래를 잘 예측하는 것.

Basic of deeplunning algorithm

RNN



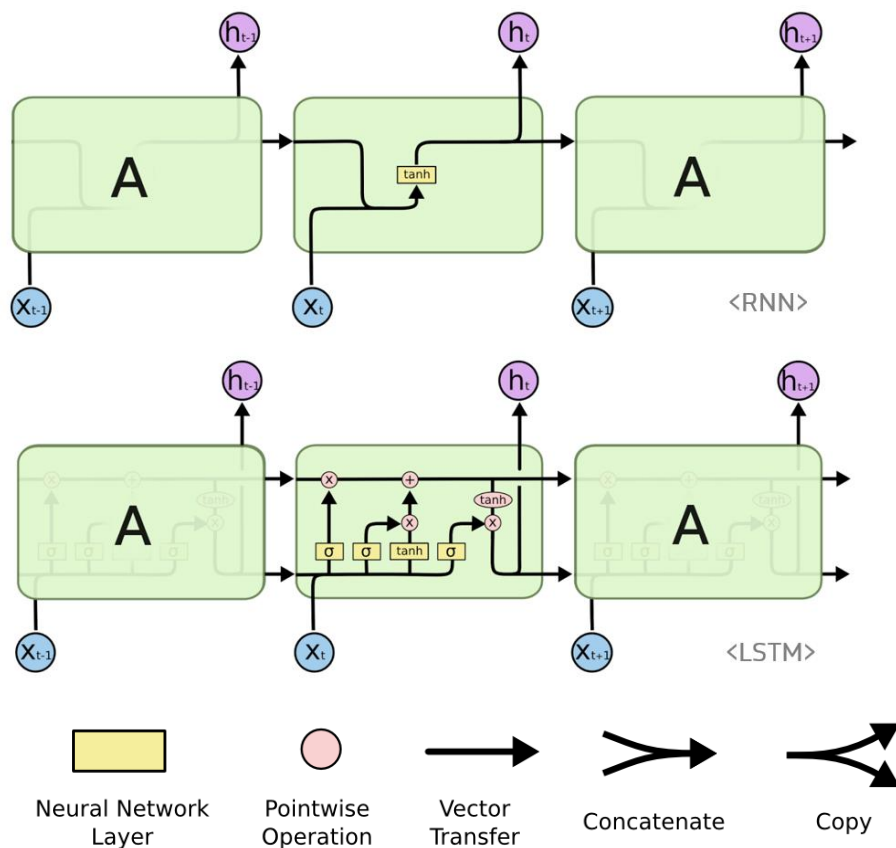
hidden 노드가 방향을 가진 엣지로 연결돼 순환 구조를 이루는(directed cycle) 인공신경망의 한 종류이다. 순차적으로 등장하는 데이터 처리에 적합한 모델이다.

RNN은 데이터가 충분히 wide하거나 deep 하지 않다면 긴문장을 제대로 예측하지 못한다.

RNN로 sequential data를 분류하는 모델을 만들수 있으나 vanishing gradient 문제가 발생하게 된다

Basic of deeplunning algorithm

- LSTM



RNN 은닉 층의 메모리셀에 망각 게이트, 입력 게이트, 출력 게이트를 추가된 모델.

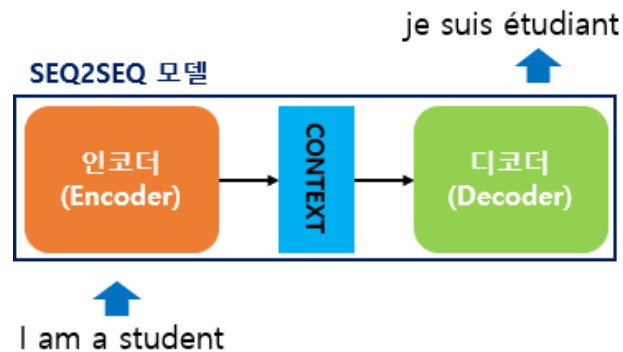
Forget gate를 이용해 기억을 장, 단기적으로 저장할 수 있어서 장단기 메모리라고 한다.

LSTM은 vanishing gradient problem을 피하기 위해 설계되었다.

하지만 여전히 Vanishing gradient problem을 가지고있고, 히든 레이어의 구조가 복잡하다.

Basic of deeplunning algorithm

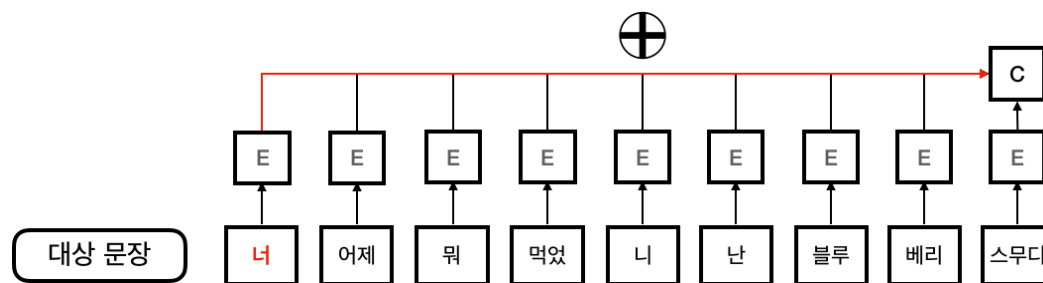
- Seq2Seq



Seq2Seq 모델은 시퀀스 길이와 순서를 자유롭게 한다. 입력된 시퀀스로부터 다른 도메인의 시퀀스를 출력하는 다양한 분야에서 사용된다. 전체 input을 살펴본 후, 임의의 Context Vector(문맥 벡터)를 출력한다.

Basic of deeplunning algorithm

- Attention



attention mechanism이란 입력된 벡터의 가중치를 부여해서 매 예측 시점마다 이를 반영하여 디코더의 출력 단어를 예측한다.

바로 직전 입력 뿐만 아니라 다른 모든 입력들이 현재결과에 얼마나 기여하는지 가중치의 형태로 나타낸다.

Attention의 본질적인 아이디어란 **“입력간 상관관계를 가중치의 형태로 모델에 반영한다”**이다

Basic of deeplunning algorithm

- Transformer

RNN을 사용하지않고 입력을 병렬화하여 단순히 행렬곱으로 한번에 연산한다. 한번의 연산으로 모든 중요 정보는 각 단계 인코딩된다.

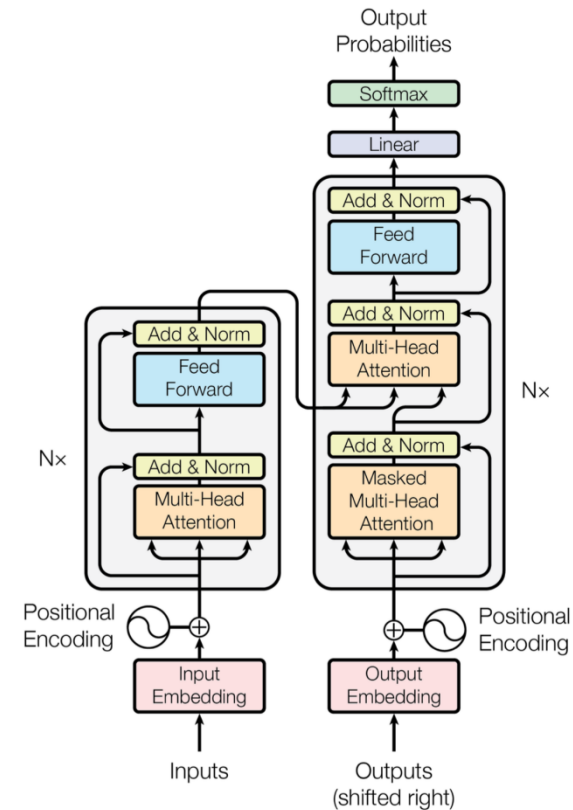
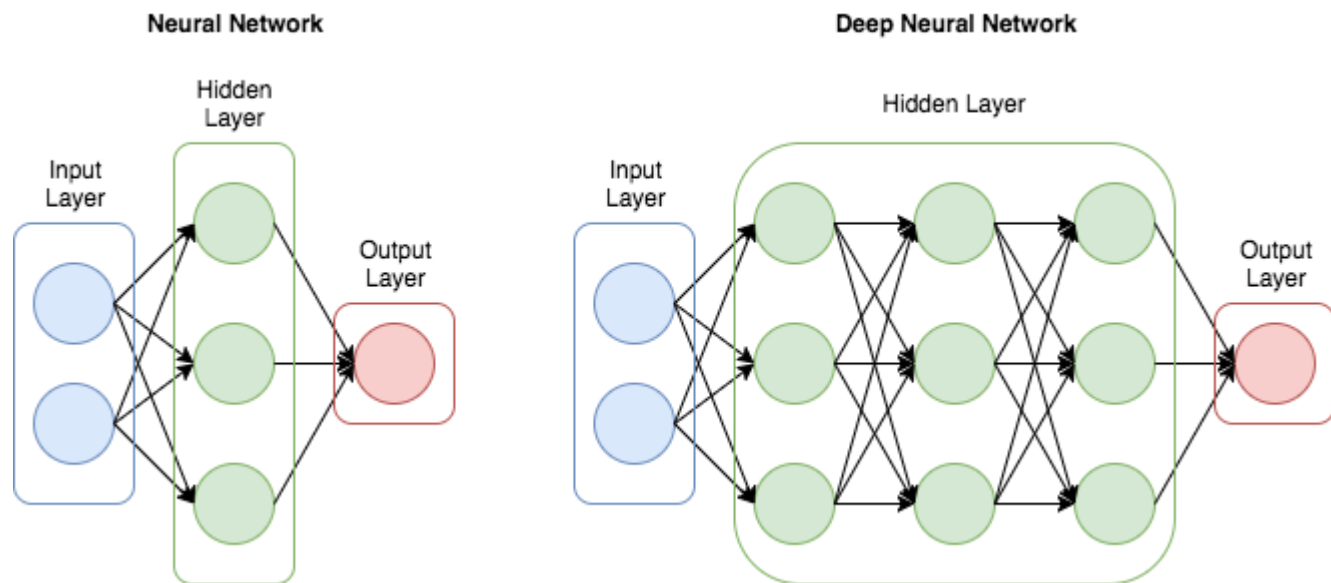


Figure 1: The Transformer - model architecture.

Basic of deeplunning algorithm

- DNNs



모델 내 은닉층을 많이 늘려서 학습의 결과를 향상시키는 방법이 등장하였고 이를 DNN(Deep Neural Network)라고 한다. DNN은 은닉층을 2개 이상 지닌 모든 종류의 학습 방법.

Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting


Bryan Lim^{a,1,*}, Sercan O. Arı^k^b, Nicolas Loeff^b, Tomas Pfister^b

^aUniversity of Oxford, UK


^bGoogle Cloud AI, USA

Tasks

 Edit

 Interpretable Machine Learning


 Time Series

 Time Series Forecasting

[Submitted on 19 Dec 2019 ([v1](#)), last revised 27 Sep 2020 (this version, v3)]


Code

 Edit

 [google-research/google-research](#)

★ 17,804

 TensorFlow

 [awslabs/gluon-ts](#)

★ 1,917

 mxnet

 [jdb78/pytorch-forecasting](#)

★ 1,058

 PyTorch

 [mattsherar/Temporal_Fusion_Transform](#)

★ 102

 PyTorch



★ 53

 PyTorch

[dehoyosb/temporal_fusion_transformer_pytorch](#)

[See all 15 implementations](#)

TFT : Background

Multi-horizon forecasting

다중 시점 예측은 여러 미래 time step에서 관심변수를 예측.

one step forecasting과 달리 사용자에게 전체 루트에 대한 추정값의 엑세스를 제공해서 향후 여러단계에서 작업을 최적화 할 수 있게 돕는다.

BUT! 다중 시점 예측은 데이터(알려진 정보, 외인적 시계열, 정적 메타데이터 등)의 이질성과 상호작용에 대한 정보가 거의 없기 때문에 특히 어렵다.

TFT : Background

Deep neural networks(DNNs)의 개선 방향은 attention기반 방법을 사용한 transformer 모델을 구성하는 것이다. 하지만 이 모델에는 여러 문제가 존재.

Problems

1. Multi-horizon 예측에 일반적으로 존재하는 다양한 유형의 입력을 고려하지 못한다.
2. 모든 외인적 입력이 미래에 알고있다고 가정한다.
3. 각 단계에서 다른 시간에 종속되어있는 기능의 중요한 정적 공변락을 무시한다.
4. 현재 대부분의 아키텍처는 'black box'모델이기 때문에 모델이 예측에 도달하는 방법을 설명하기 어렵고, 사용자가 모델을 신뢰하고 디버깅하는데 어렵다.
5. 기존DNN의 사후방법은 입력기능의 시간 순서를 고려하지않기때문에 일반적으로 사용되는 explainability(설명력 높은) methods는 시계열에 적용하기 적합하지 않다.
6. Multi-horizo예측에는 언어, 음성뿐만 아니라 다양한 유형의 입력기능이 있다. 그래서 관련된 time steps에 대한 통찰력을 제공 할 순 있지만, 주어진 time steps에서 다른 featur의 중요성을 알아 낼 수는 없다.

TFT : Background

TFT's IDEA

1. 정적 공변량 인코더를 통합한다. 네트워크의 다른 부분에서 사용하기 위해 컨텍스트 벡터를 인코딩.
2. 관련없는 입력의 기여를 최소화하기 위해 전체 gating 매커니즘 및 sample 종속 변수를 선택한다.
3. 알려진, 혹은 관찰된 입력을 로컬로 처리하는 Seq2Seq 계층을 사용한다
4. 데이터 세트 내에 존재하는 임의의 장기 종속성을 학습하기 위해 일시적으로 self-attention 디코더를 사용한다.

Point! 미래 시점에서 미리 알 수 있는 정보들을 활용할 수 있도록 딥러닝 모델을 구성

TFT : Background

TFT's interpretability(해석 가능성) use cases

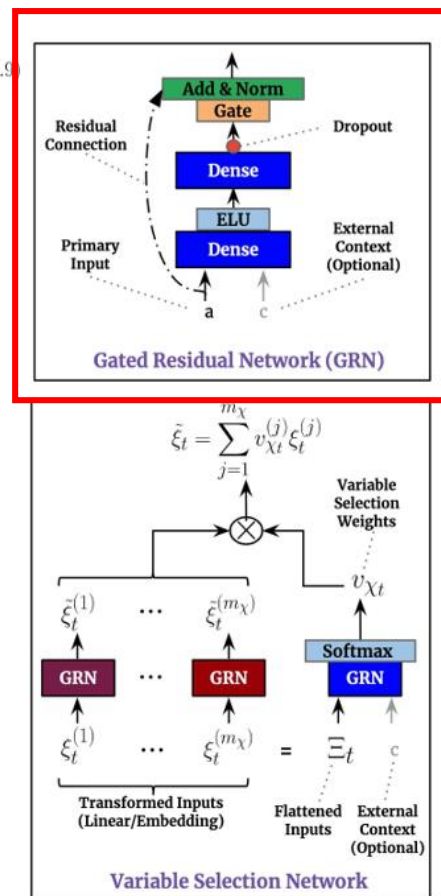
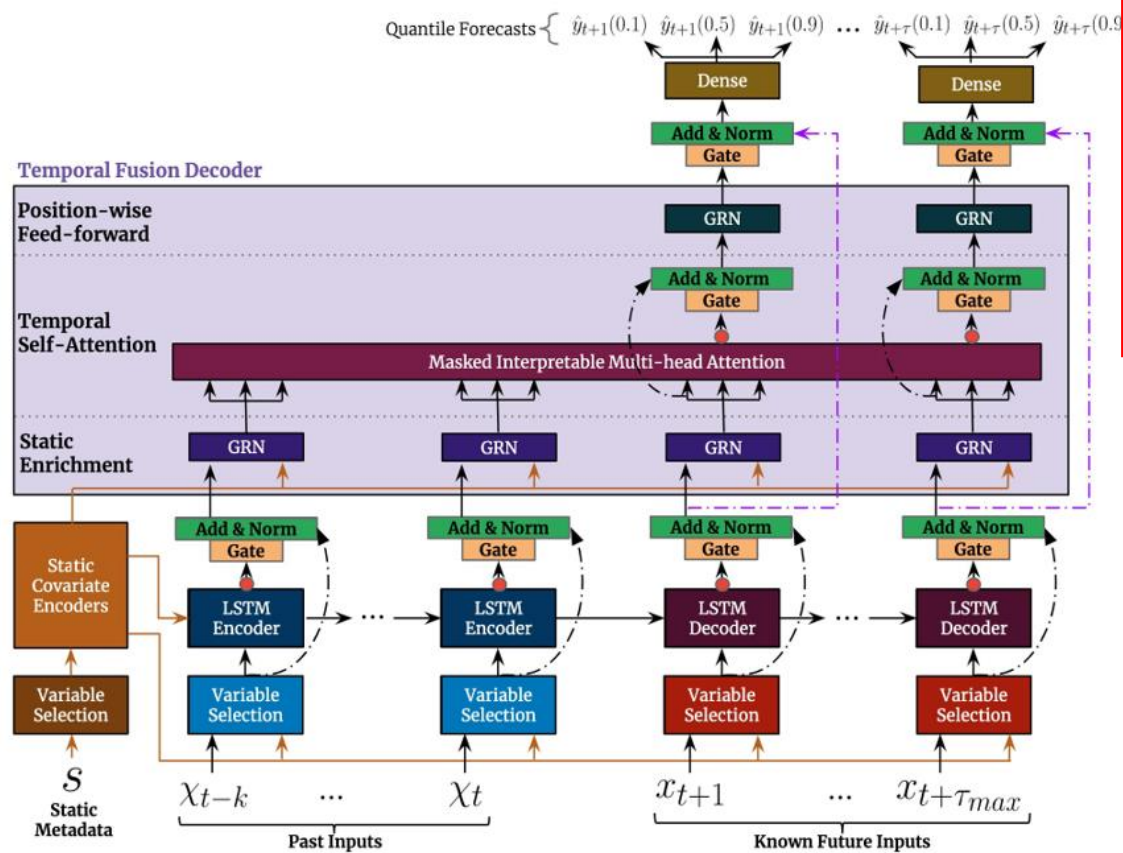
1. 예측 문제에 대해 전반적으로 중요한 변수 식별
2. 지속적인 시간 패턴 식별
3. 중요한 이벤트 식별



사용자가 모델을 이해하고, 디버깅할 수 있다.

알고리즘만

TFT 아키텍처



전체적으로 Gating Mechanisms(게이팅 메커니즘)을 통해 네트워크 복잡성을 해결하려는 방식이 눈에 띈다.

TFT모델링에 사용된 분석 알고리즘

1. Gating Mechanisms
2. 변수 선정 Network
3. 정적 공변량 Encoder
4. Interpretable Multi-Head Attention
5. Temporal Processing
6. Quantile Outputs, 예측 시점 구간

게이팅 매커니즘

모델 학습 시 데이터중 불필요한 시점의 입력을 통제하여 장기 예측을 유리하게 한다.

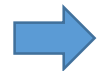
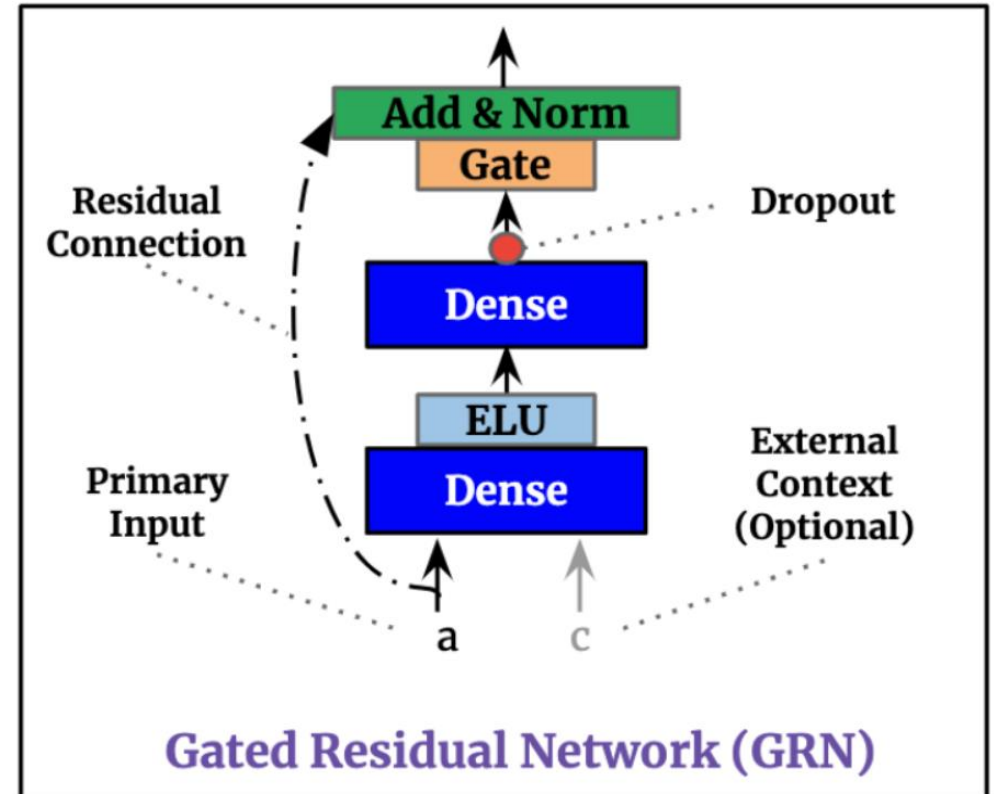
입력 시퀀스 중 유용한 정보를 가지고있는 시점을 가려내기 위해 GLU를 적용한 GRN 기법 활용.

게이팅 매커니즘

GRN

TFT의 대부분의 레이어에서 적용되는 핵심 알고리즘.

Residual Network를 사용하여 과적합을 방지하며, Layer의 표준화하고, 게이트 통과 결과값을 Add한다.



장기 시퀀스에서 필요한 정보를 모델에 입력하도록 유도, 입력 시퀀스중 유용한 정보를 가진 시점을 가려낸다.

변수 선정 Networks

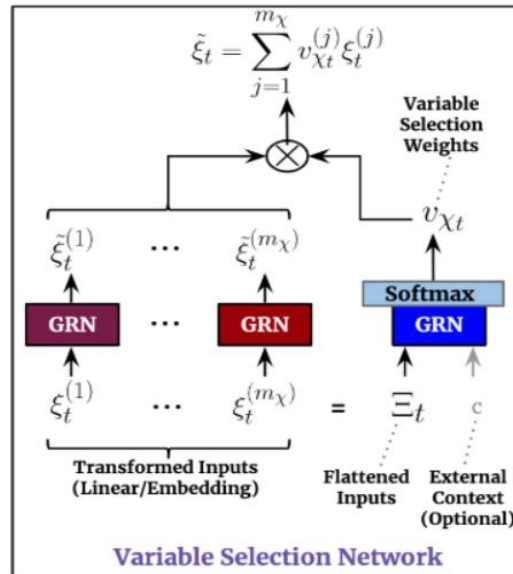
TFT에선 각 time step 마다 적절한 입력 변수를 선정한다.

Why? 시계열 예측 분석에서 변수를 선정하는 과정은 모델 성능 향상에 큰 효과를 미칠 수 있기때문에.

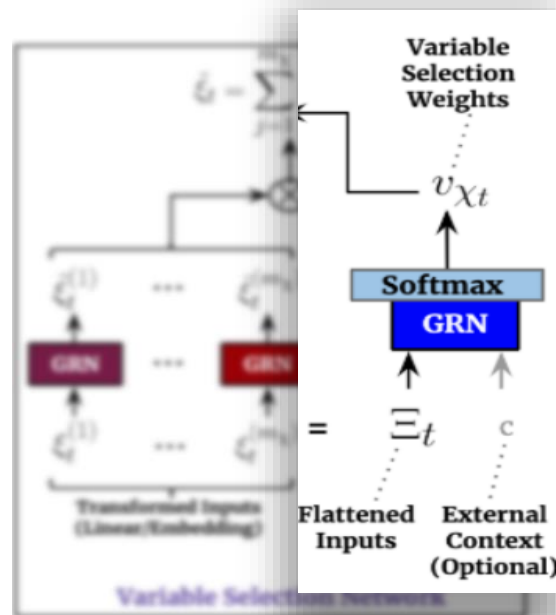
-> instance-wise 변수 선정 네트워크 기법

변수 선정 Networks

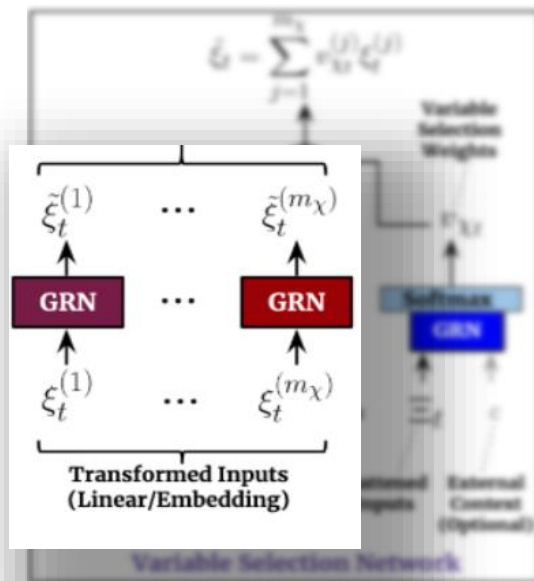
- instance-wise 변수 선정 네트워크



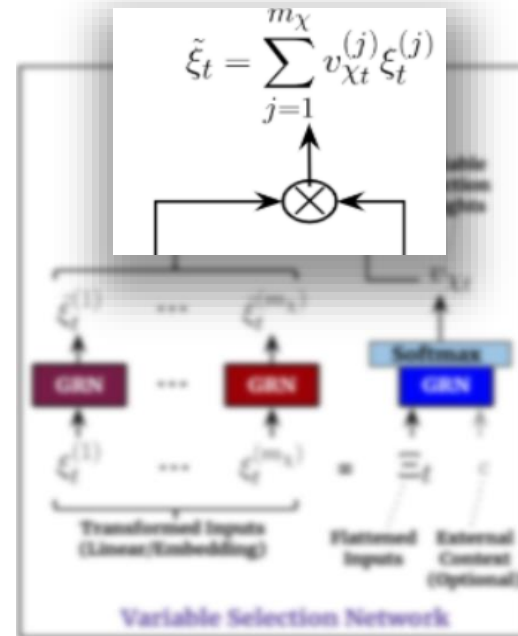
변수 선정 Networks



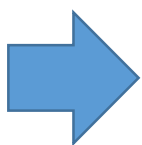
통합된 전체 입력 변수, 외부 context vector를 GRN에 입력 후 Softmax를 통해 변수를 선정한다.



추가적으로, 전체 입력 변수 각각에 대해서도 GRN모형을 적용

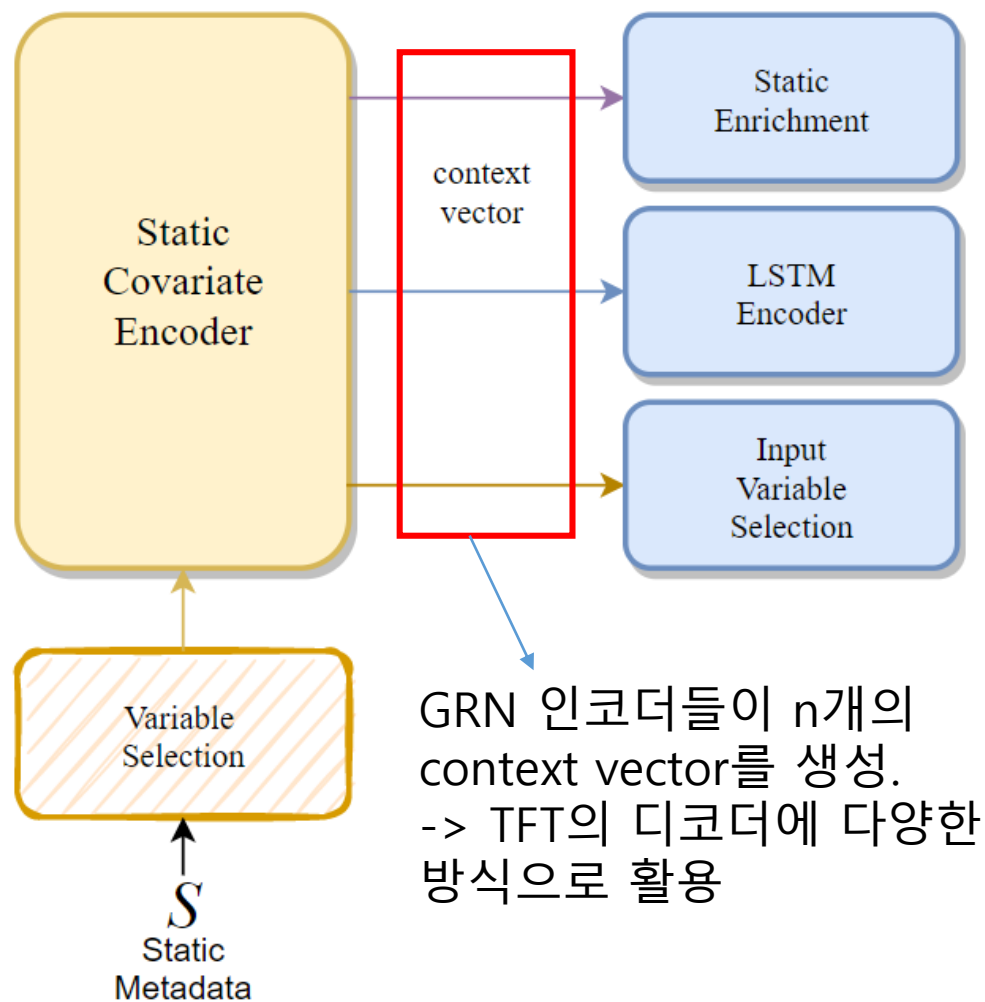


앞의 GRN 모델을 통합하여 t시점의 활용 변수를 선정한다.



딥러닝 모델의 잡음을 효과적으로 제어하여 모델의 성능을 높인다.

정적 공변량 인코더



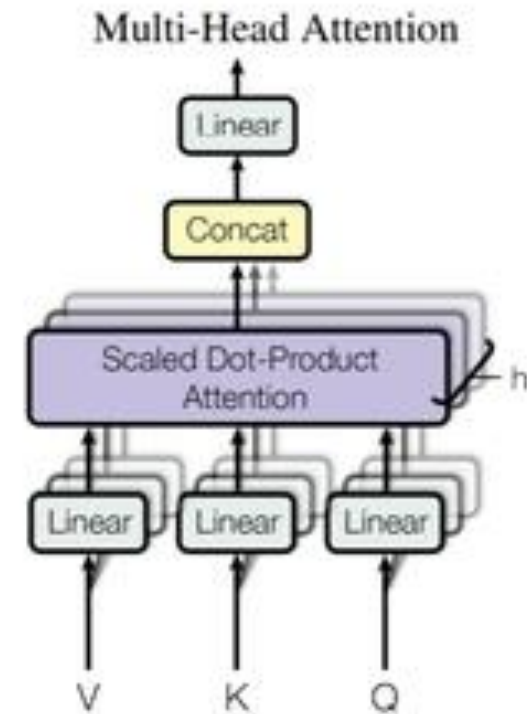
GRN 인코더들이 n 개의 context vector를 생성.
-> TFT의 디코더에 다양한 방식으로 활용

정적 공변량을 모델에 입력 할 수 있는 형태로 통합하여, 여기서 만들어진 context vector를 동적 변수의 조건으로 설정하게 한다.
이 벡터들은 각각 GRN을 통해 모델에 인코딩 되는데, GRN 가중치는 전체 계층에서 공유된다.

Interpretable Multi-Head Attention

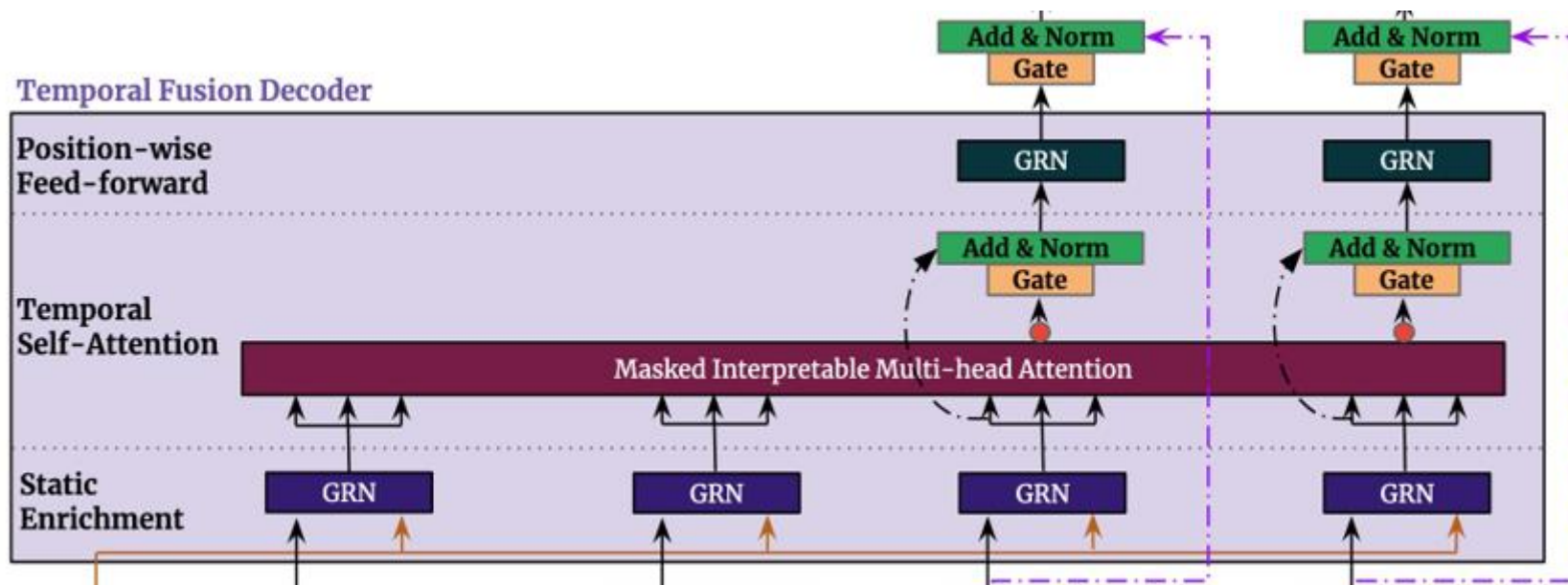
TFT 에서는 장기간 time step을 학습시키기 위해 트랜스포머 기반의 multi-head attention 기법으로 설명력을 강화하는 새로운 attention 가중치 계산방식을 만들었다.

각 head가 서로 다른 time 패턴을 학습할 수 있다. 또한 attention 가중치 값을 공유하는 방식으로 특성의 중요도를 추출한다.



Temporal Processing

관측 (observed) 변수 및 알고있는 변수를 장단기 예측에 input 하기 위해, 시점처리와 multi-head attention 기법을 적용했다.

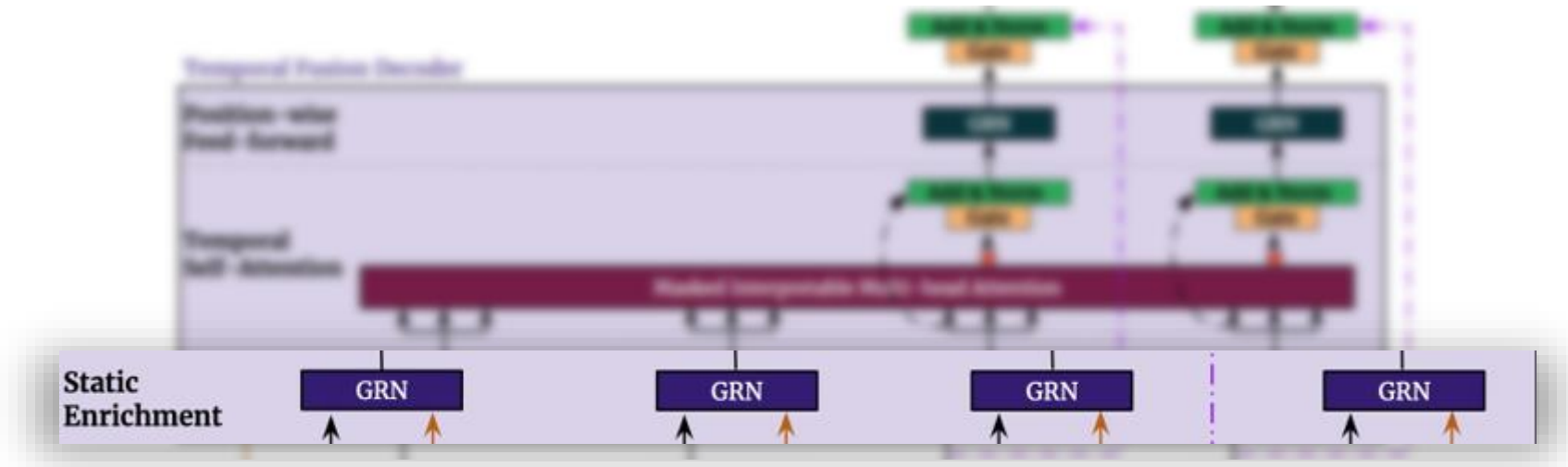


Temporal Processing

(1) Locality Enhancement with Seq2Seq Layer

TFT 모델은 LSTM 인코더, 디코더 구조를 통해 locality를 강화한다. 목표값의 해당 시점의 로컬 컨텍스트를 활용하면 attention 기반 모델의 성능이 향상된다. TFT 모델의 입력값은 출력값의 길이와 다르기 때문에 Seq2Seq 모델 방식으로 접근한다.

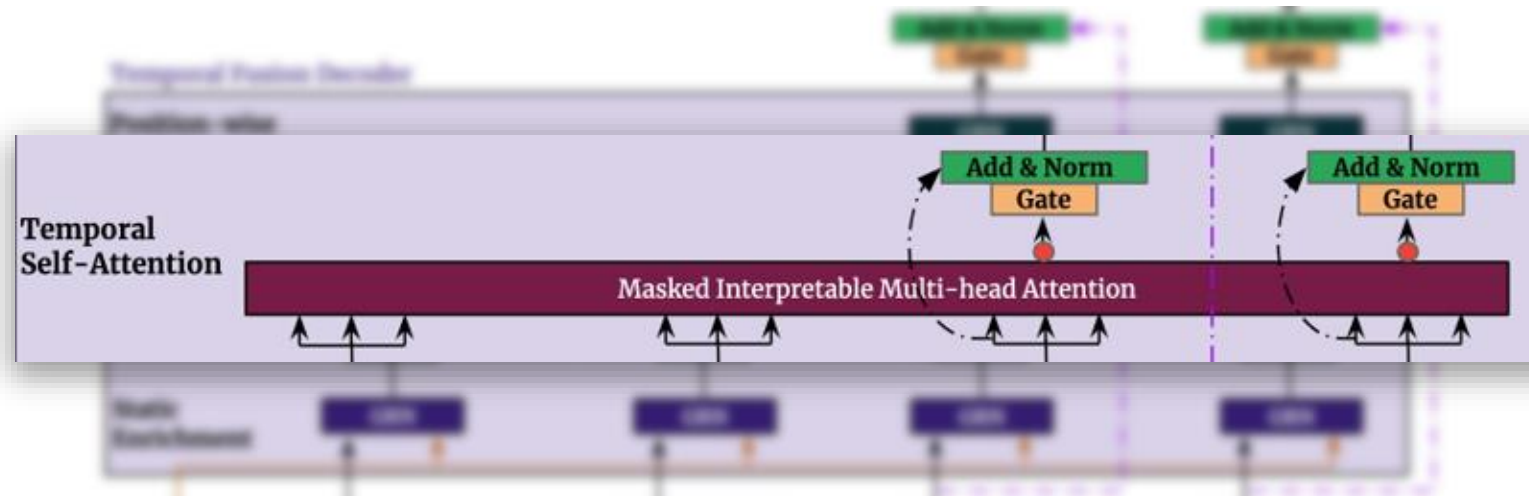
Temporal Processing



(2) Static Enrichment Layer

고정변수의 영향이 전체 예측 모델에 중요한 역할을 할 수도 있기 때문에 이를 반영하고자 TFT는 고정변수의 context vector를 GRN 모델을 통해 전체 레이어에 전달한다.

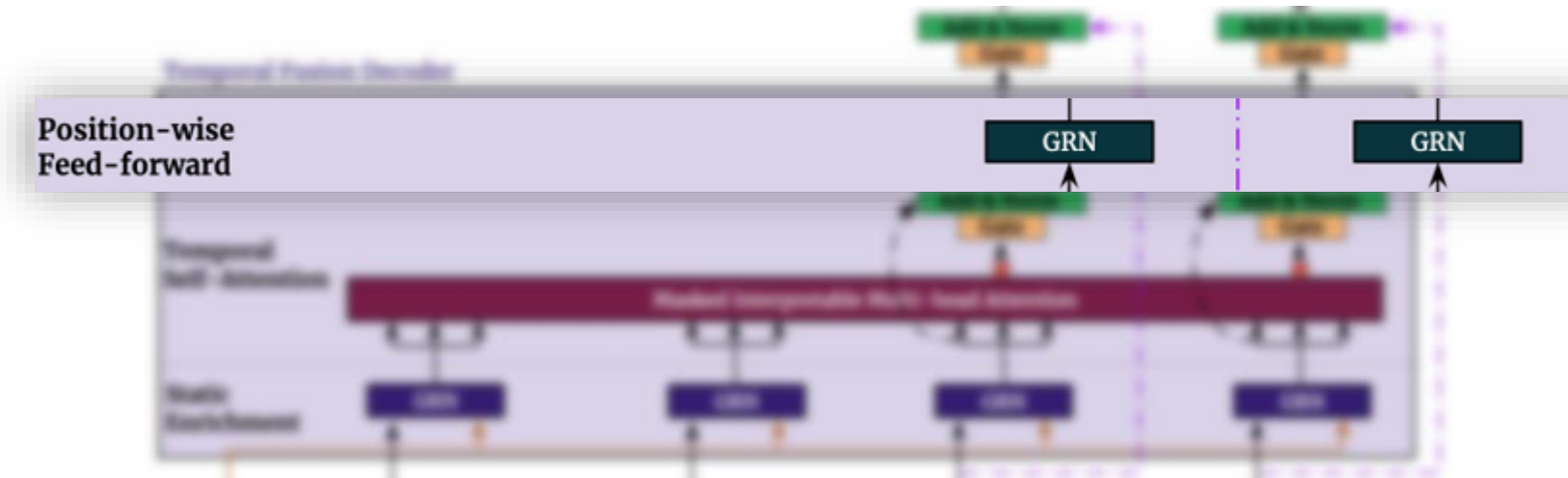
Temporal Processing



(3) Temporal Self-Attention Layer

Static Enrichment 과정 후, self-attention을 적용. 디코더 시점에서 masking을 수행하여 temporal 차원이 특성에만 집중하도록 한다. 또한 인과적인 정보 흐름을 보존하고 장거리 시점의 종속성을 제대로 포착하기때문에 장기간 예측에 유리하다.

Temporal Processing

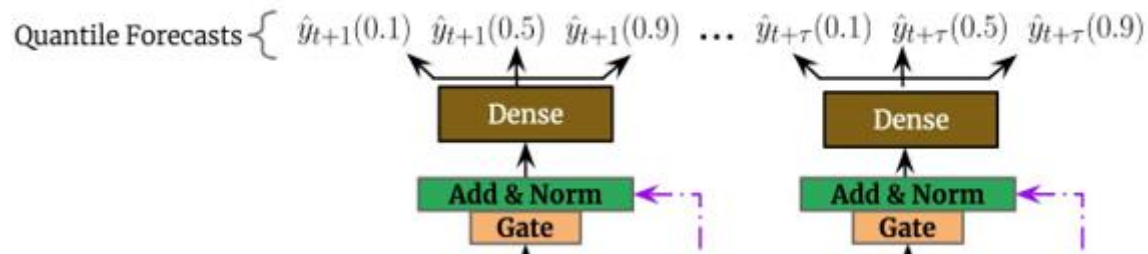


(4) Position-wise Feed-Forward Layer

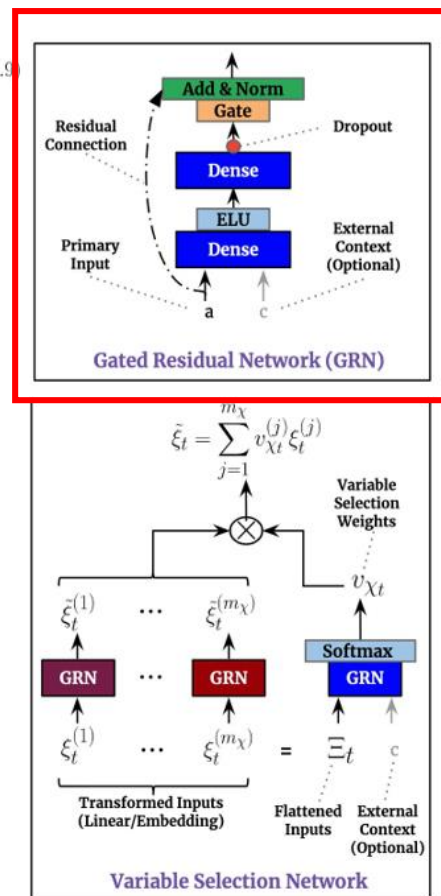
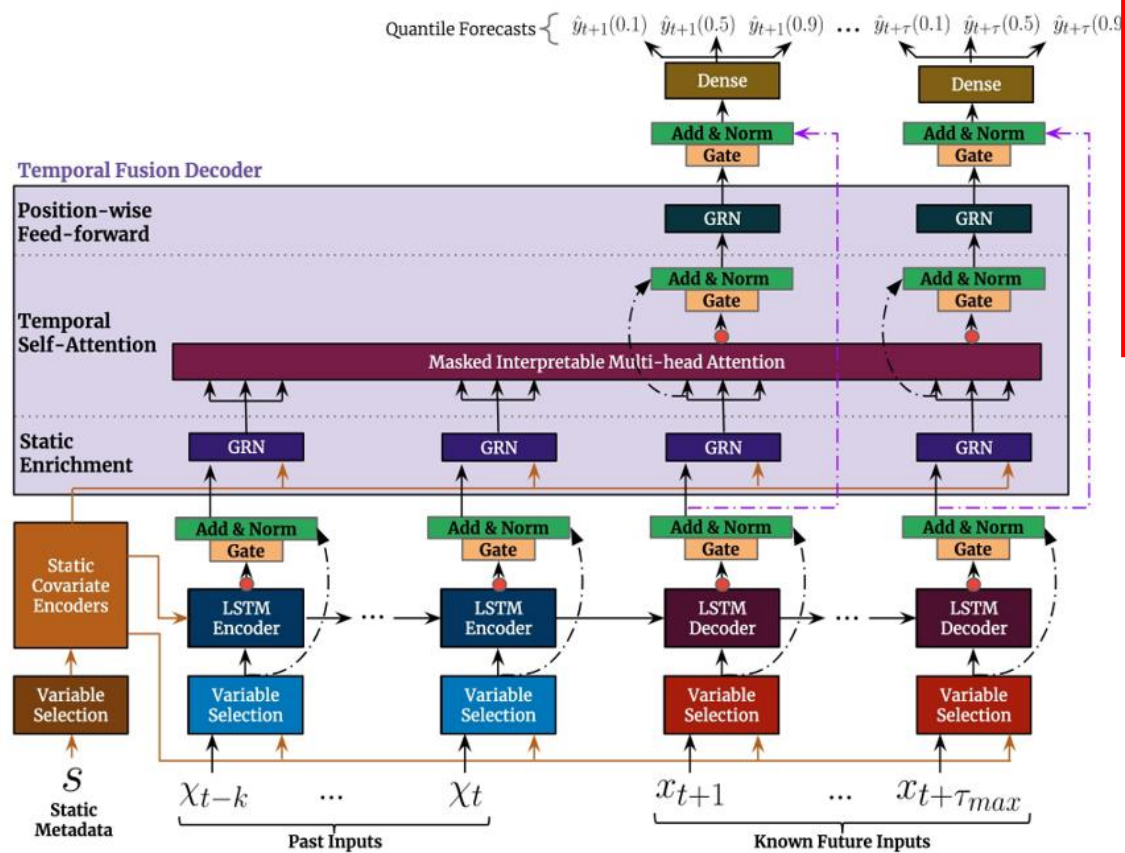
TFT 모델에서는 Self-attention layer에 GRN으로 구성된 추가적인 비선형 레이어 프로세스를 추가한다. 이 레이어는 전체 레이어구조에 가중치가 전파된다. 또한, 전체 트랜스포머 레이어와 Seq2Seq레이어를 직접 연결하는 경로를 설정해, 모델의 복잡성 문제를 해결했다.

Quantile Outputs , 예측 시점 구간

각 예측 시점마다 목표 변수의 범위를 결정하기 위한 분위수 예측.
TFT는 예측 시점에 예측 간격을 생성하는 방식을 사용한다. 매 시점마다 10% 분위수, 50% 분위수, 90% 분위수를 동시에 예측한다.
예측은 미래의 시점에 대해서만 생성된다.



TFT 아키텍처



전체적으로 Gating Mechanisms(게이팅 메커니즘)을 통해 네트워크 복잡성을 해결하려는 방식이 눈에 띈다.

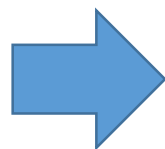
Computational Cost

- 기준: NVIDIA Tesla V100 GPU

Table 1: Information on dataset and optimal TFT configuration.

	Electricity	Traffic	Retail	Vol.
<u>Dataset Details</u>				
Target Type	\mathbb{R}	[0, 1]	\mathbb{R}	\mathbb{R}
Number of Entities	370	440	130k	41
Number of Samples	500k	500k	500k	~100k
<u>Network Parameters</u>				
k	168	168	90	252
τ_{max}	24	24	30	5
Dropout Rate	0.1	0.3	0.1	0.3
State Size	160	320	240	160
Number of Heads	4	4	4	1
<u>Training Parameters</u>				
Minibatch Size	64	128	128	64
Learning Rate	0.001	0.001	0.001	0.01
Max Gradient Norm	0.01	100	100	0.01

Optimal TFT model
Training
6 hours



Optimal TFT model
Validation 추론
8 minute

TFT는 좋은 컴퓨팅 리소스 없이도 배포할 수 있다.

Benchmarks

다중 시점 예측을 위한 다양한 모델과 광범위하게 비교.

간단한 데이터 세트는 ARIMA, ETS, TRMF도 포함해서 벤치마킹했다.

-> 다양한 데이터 세트에 대해 다른 모델보다 더 훌륭한 성능을 보인다.

	ARIMA	ETS	TRMF	DeepAR	DSSM
Electricity	0.154 (+180%)	0.102 (+85%)	0.084 (+53%)	0.075 (+36%)	0.083 (+51%)
Traffic	0.223 (+135%)	0.236 (+148%)	0.186 (+96%)	0.161 (+69%)	0.167 (+76%)
	ConvTrans	Seq2Seq	MQRNN	TFT	
Electricity	0.059 (+7%)	0.067 (+22%)	0.077 (+40%)	0.055*	
Traffic	0.122 (+28%)	0.105 (+11%)	0.117 (+23%)	0.095*	

(a) P50 losses on simpler univariate datasets.

	ARIMA	ETS	TRMF	DeepAR	DSSM
Electricity	0.102 (+278%)	0.077 (+185%)	-	0.040 (+48%)	0.056 (+107%)
Traffic	0.137 (+94%)	0.148 (+110%)	-	0.099 (+40%)	0.113 (+60%)
	ConvTrans	Seq2Seq	MQRNN	TFT	
Electricity	0.034 (+26%)	0.036 (+33%)	0.036 (+33%)	0.027*	
Traffic	0.081 (+15%)	0.075 (+6%)	0.082 (+16%)	0.070*	

(b) P90 losses on simpler univariate datasets.

	DeepAR	CovTrans	Seq2Seq	MQRNN	TFT
Vol.	0.050 (+28%)	0.047 (+20%)	0.042 (+7%)	0.042 (+7%)	0.039*
Retail	0.574 (+62%)	0.429 (+21%)	0.411 (+16%)	0.379 (+7%)	0.354*

(c) P50 losses on datasets with rich static or observed inputs.

	DeepAR	CovTrans	Seq2Seq	MQRNN	TFT
Vol.	0.024 (+21%)	0.024 (+22%)	0.021 (+8%)	0.021 (+9%)	0.020*
Retail	0.230 (+56%)	0.192 (+30%)	0.157 (+7%)	0.152 (+3%)	0.147*

(d) P90 losses on datasets with rich static or observed inputs.

제안된 아키텍처의 기여도 이점을 정량화하기위해 제거분석을 수행한다. 아키텍처 대비 손실 증가율을 정량화한다.

Ablation Analysis

제안된 아키텍처의 기여도 이점을 정량화하기위해 제거분석을 수행한다.

아키텍처 대비 손실 증가율을 정량화한다

temporal relationships	P90손실이 평균 6% 이상, 일부 데이터 세트에선 20%이상 증가
local processing	
self-attention layers	
정적 공변량 인코더	P90 손실이 2.6~ 4.1% 증가한다
인스턴스 별 변수선택	
Gating	P90 손실이 평균 1.9% 증가

Interpretability Use Cases

세가지 해석가능성 Use Cases

1. Analyzing Variable Importance. 예측에서 각 입력 변수의 중요성
2. Visualizing Persistent Temporal Patterns. 지속적인 시간 패턴 시각화
3. Identifying Regimes & Significant Events. 중요한 변화를 만드는 event 식별

Interpretability Use Cases

1. Analyzing Variable Importance

먼저 섹션에 설명 된 변수 선택 가중치를 분석하여 변수 중요도를 정량화한다.

가중치가 0.1 보다 큰 값은 자주색으로 강조표시된다.

	10%	50%	90%
Item Num	0.198	0.230	0.251
Store Num	0.152	0.161	0.170
City	0.094	0.100	0.124
State	0.049	0.060	0.083
Type	0.005	0.006	0.008
Cluster	0.108	0.122	0.133
Family	0.063	0.075	0.079
Class	0.148	0.156	0.163
Perishable	0.084	0.085	0.088

	10%	50%	90%
Transactions	0.029	0.033	0.037
Oil	0.062	0.081	0.105
On-promotion	0.072	0.075	0.078
Day of Week	0.007	0.007	0.008
Day of Month	0.083	0.089	0.096
Month	0.109	0.122	0.136
National Hol	0.131	0.138	0.145
Regional Hol	0.011	0.014	0.018
Local Hol	0.056	0.068	0.072
Open	0.027	0.044	0.067
Log Sales	0.304	0.324	0.353

(a) Static Covariates

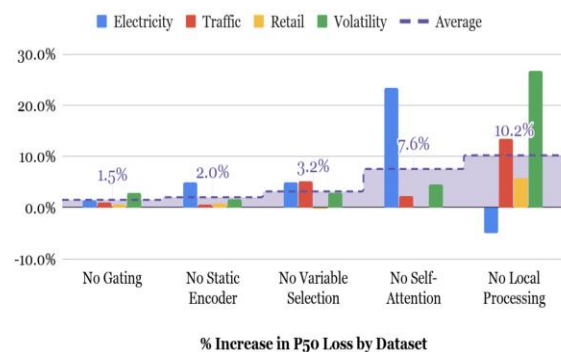
(b) Past Inputs

	10%	50%	90%
On-promotion	0.155	0.170	0.182
Day of Week	0.029	0.065	0.089
Day of Month	0.056	0.116	0.138
Month	0.111	0.155	0.240
National Hol	0.145	0.220	0.242
Regional Hol	0.012	0.014	0.060
Local Hol	0.116	0.151	0.239
Open	0.088	0.095	0.097

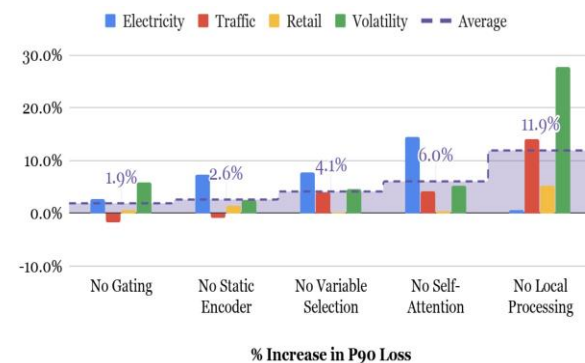
(c) Future Inputs

Interpretability Use Cases

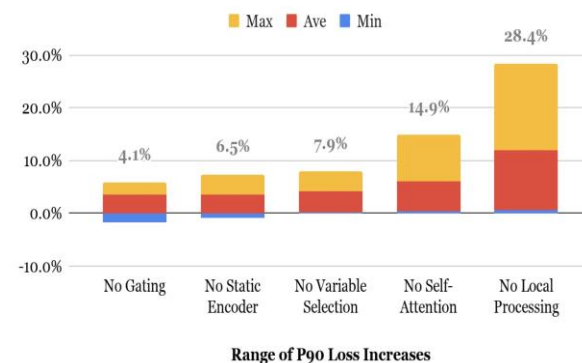
1. Analyzing Variable Importance: Ablation Analysis



(a) Changes in P50 losses across ablation tests



(a) Changes in P90 losses across ablation tests



P50, P90에 대하여 전반적으로 더 많은 Loss가 발생했다.

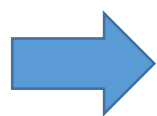
Interpretability Use Cases

2. Visualizing Persistent Temporal Patterns

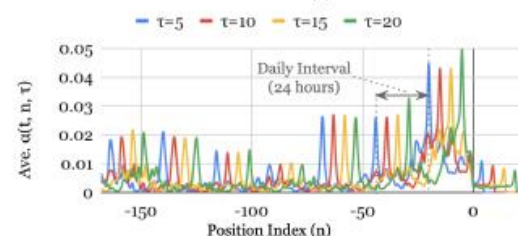
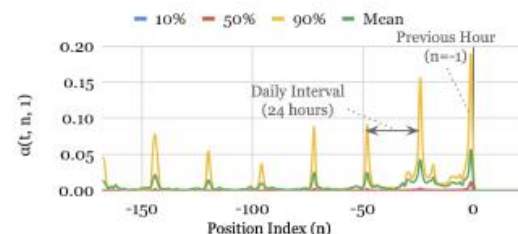
모든 테스트 데이터 세트에 대한 attention 가중치 패턴을 보여준다.

계절성 및 지연 분석을 위해 모델 기반 사양에 의존하는 다른 기존 및 기계 학습 시계열 방법과 달리 TFT는 가공하지 않은 학습 데이터에서 이러한 패턴을 학습 할 수 있다.

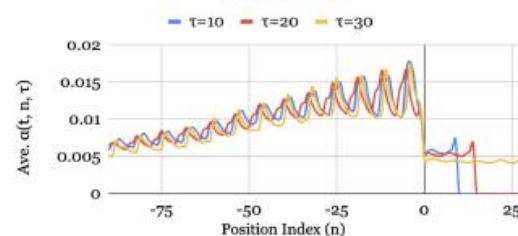
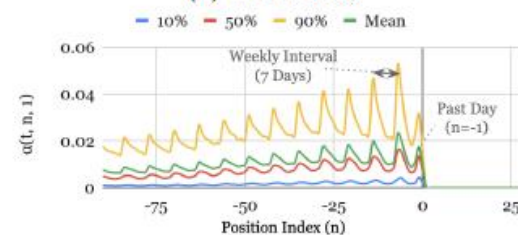
TFT는 인간의 어려운 코딩없이 가공하지 않은 훈련 데이터에서 이러한 지속적인 시간 패턴을 학습함.



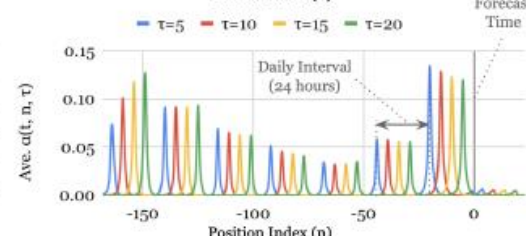
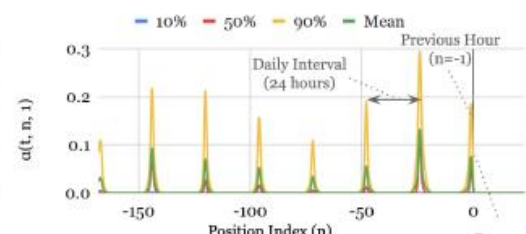
전문가와의 신뢰 구축에 매우 유용. 특정 기능 엔지니어링 또는 데이터 수집을 통해 모델 개발자는 이를 모델 개선에 사용할 수도 있음.



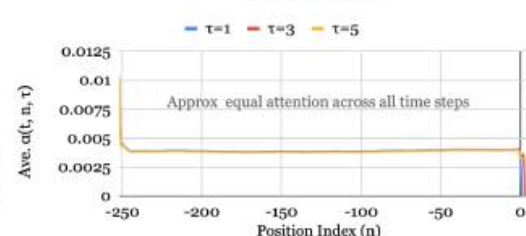
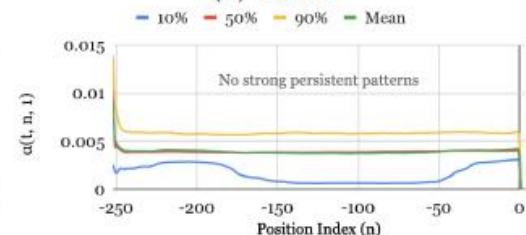
(a) Electricity



(c) Retail



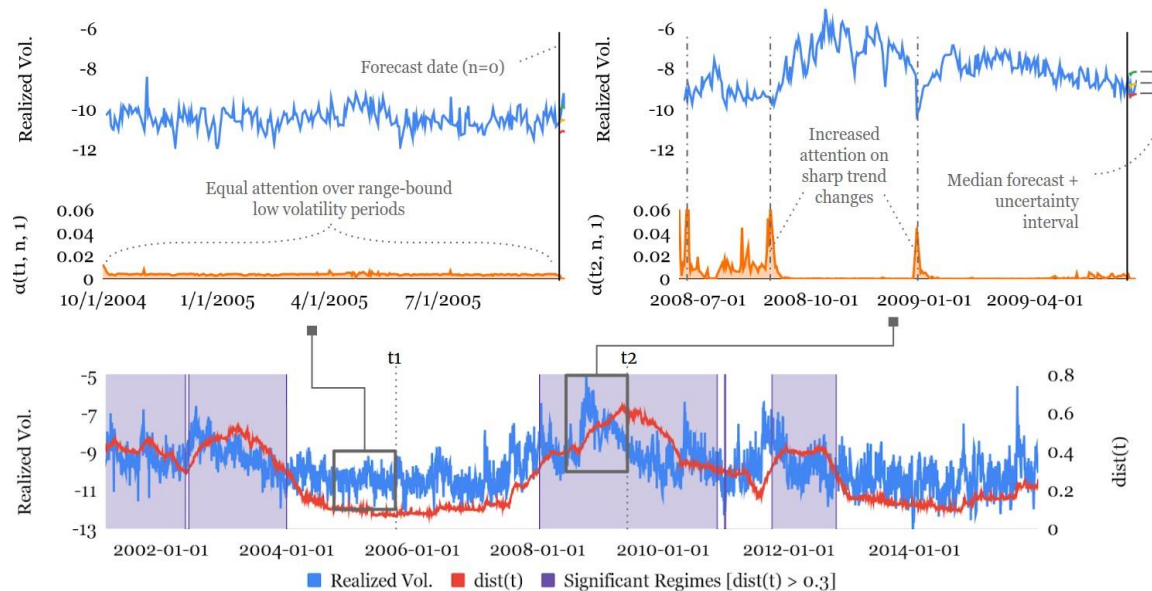
(b) Traffic



(d) Volatility

Interpretability Use Cases

3. Identifying Regimes & Significant Events



중요한 영역을 자주색으로 강조표시.

변동성이 낮을 때 과거 입력에 동일한 attention을 두는 반면에 변동성이 높은 기간에는 급격한 추세 변화에 더 많은 attention을 준다는 것을 확인 할 수 있다.

Reference

- [1] B. Lima, S. Arkb, N. Loeb, T. Psterb , Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting - arxiv.org/abs/1912.09363
- [2] Ashish V, Noam Sr, Niki P, Jakob U, Llion J, Aidan N. Gomez, Lukasz K, Illia P, : Attention Is All You Need - <https://arxiv.org/abs/1706.03762>
- [2] Lee. H.S SOTA algorithm review - bigwaveai.tistory.com/5
- [3] <https://wikidocs.net/24996>
- [4] L. Mao, Gated Linear Units(GLU) and Gated CNN- <https://leimao.github.io/blog/Gated-Linear-Units/>
- [5] colah ,Understanding LSTM Networks - <http://colah.github.io/posts/2015-08-Understanding-LSTMs/>
- [6] kim. Ki, Introduction to Deep Time-series Anomaly Detection -<https://kh-kim.github.io/blog/2020/06/24/time-series-ad.html>
- [7] Arvid Kingl, TFT - <https://www.youtube.com/watch?v=M7O4VqRf8s4>