EFFECTIVE SELF –SUPERVISED TRANSFORMERS FOR SPARSE TIME SERIES DATA

Final Project Presentation

Data Science under Uncertainty Seminar

UQ Chair RWTH Aachen

Supervisor: Dr. Anamika Pandey

Oleksandr Nikolskyy

Plan

1. Introduction
2. Formal Problem Statement
3. Recap of Transformer Archtecture
4. Training Pipeline
    1. Data Preparation
    2. Masked Time Series Modelling
    3. Supervised Training
5. Optimization approach
6. Experimental Results

Introduction

- Attention-based Deep Learning Models have shown outstanding performance in a variety of structured tasks in natural language processing and computer vision.
- In this presentation I focus on the application of Self-Supervised attention-based models to sparsely sampled and irregular time-series data
- Thus, we discuss and evaluate a training pipeline consisting of multiple steps:
  - Time binning
  - Pretraining of the model learns the proper representation of each time bin
  - Supervised finetuning on a classification task
- Implementation: Pytorch + Pytorch Lightning
- Paper: **Effective Self-Supervised Transformers For Sparse Time Series Data by Labach et Al.**

Problem statement

- Given

$\mathcal{D} = \{(s^p, W^p, y^p)\}_{p=1}^N$ Dataset

$W^p = (w_1^p, .., w_{n_p}^p)$ Event sequences

$w_i^q = (l, t, x)$ Each event consitsting of event type $l \in 1..d$, time of occurance t and observed value x

$s^q \in \mathbb{R}^{d_{\text{static}}}$ Static features

- Find a good represetation $\hat{W}$
- Find an optimal predictor for $\mathbb{E}[y|s, W]$

Transformers

- Introduced by Vasmani et al. (2017)
- Built around the Multi-Head Attention Module
- Operate on sequential/structured data
  - Initially introduced for problems in NLP domain
  - Extended to Computer Vision (ViT)
  - Interesting side developments:
    - Multimodality (eg Visual Question Answering)
      - Combination of Attention signals from multiple sources

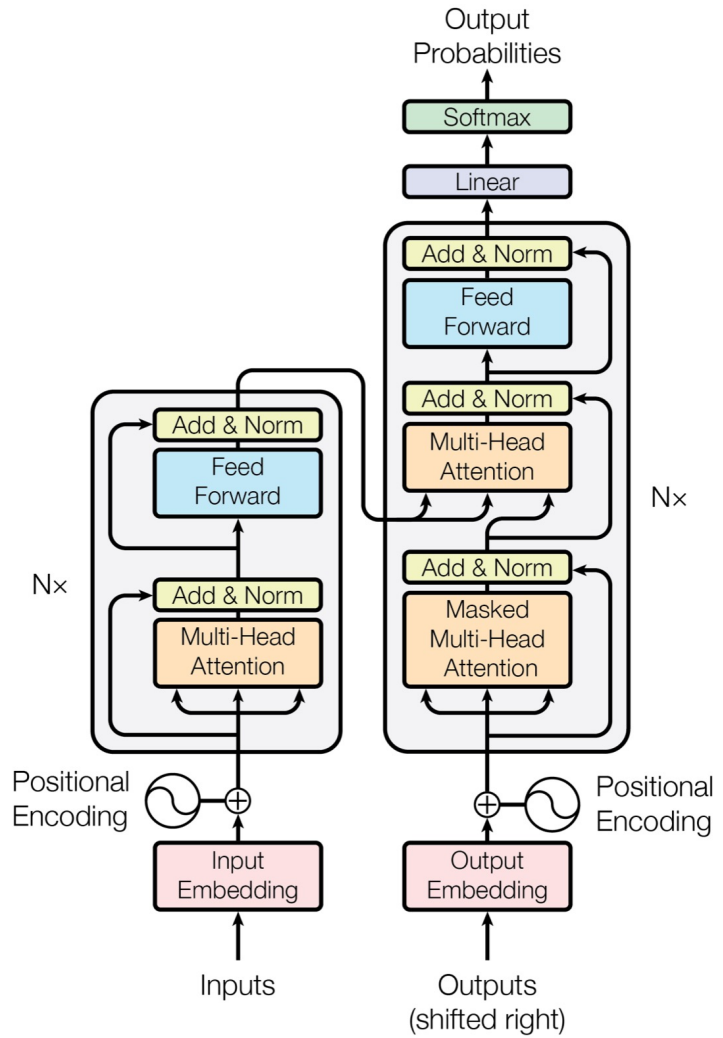


Image Source: Vaswani et al., 2017

Transfomer encoder in detail

For input Data Matrix

$$X \in \mathbb{R}^{T \times D}$$

1. Input Projection

$$\mathbb{R}^{D} \to \mathbb{R}^{D_{rep}}$$

$$X_t \mapsto \phi(W X_t)$$

$$\phi(x) := \mathrm{ReLu}(x)$$

2. Positional encoding

$$t \mapsto \phi_{\mathrm{pos}}(t)$$

ϕ_{pos} is a learned Feed Forward Network with ReLu nonlinearities

$$\phi(W X_t) + \phi_{\mathrm{pos}}(t)$$

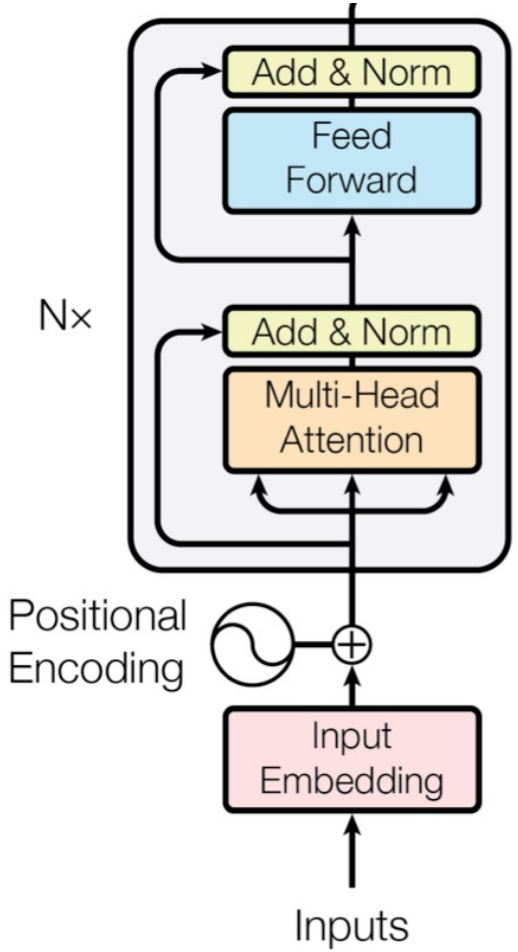


Image Source: Vaswani et al., 2017

Transfomer encoder in detail

For input Data Matrix

$$X \in \mathbb{R}^{T \times D}$$

3. Multi-Head Attention

$$Q := W_q X$$
$$K := W_k X$$
$$V := W_v X$$
$$\text{Attention}(Q, K) = \text{softmax}(\frac{QK^T}{\sqrt{D_{rep}}})$$
$$\hat{X} = \text{Attention}(Q, K)V$$

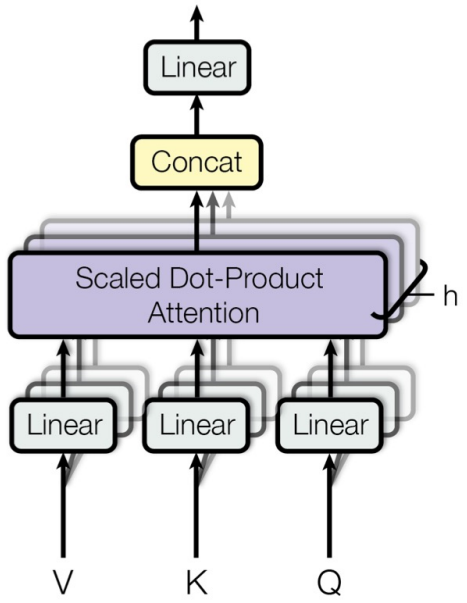


Image Source: Vaswani et al., 2017

If $A_{i,j} = \text{Attention}(X)_{i,j}$,

$$A_{i,j} = \frac{exp< (W_q X)_i, (W_k X)_j >}{C * D_{rep}}$$

Then, effectively, each dimension of the learned representation of time bin t, is a weighted sum of all time series:

$$\hat{X}_{t,d} = \sum_{i=1}^{T} A_{t,t'} X_{t',d}$$

Transfomer encoder in detail

For input Data Matrix

$$X \in \mathbb{R}^{T \times D}$$

3. Multi-Head Attention

$$Q := W_q X$$
$$K := W_k X$$
$$V := W_v X$$
$$\text{Attention}(Q, K) = \text{softmax}(\frac{QK^T}{\sqrt{D_{rep}}})$$
$$\hat{X} = \text{Attention}(Q, K)V$$

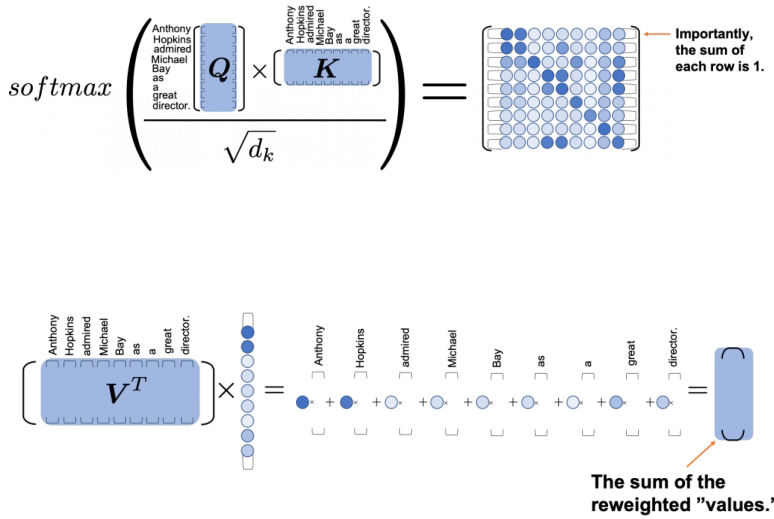


Image Source: https://wikidocs.net/167212

If $A_{i,j} = \text{Attention}(X)_{i,j}$,

$$A_{i,j} = \frac{exp< (W_q X)_i, (W_k X)_j >}{C * D_{rep}}$$

Then, effectively, each dimension of the learned representation of time bin t, is a weighted sum of all time series:

$$\hat{X}_{t,d} = \sum_{i=1}^{T} A_{t,t'} X_{t',d}$$

Transfomer encoder in detail

For input Data Matrix

$$X \in \mathbb{R}^{T \times D}$$

3. Multi-Head Attention

$$Q := W_q X$$
$$K := W_k X$$
$$V := W_v X$$
$$\text{Attention}(Q, K) = \text{softmax}(\frac{QK^T}{\sqrt{D_{rep}}})$$
$$\hat{X} = \text{Attention}(Q, K)V$$

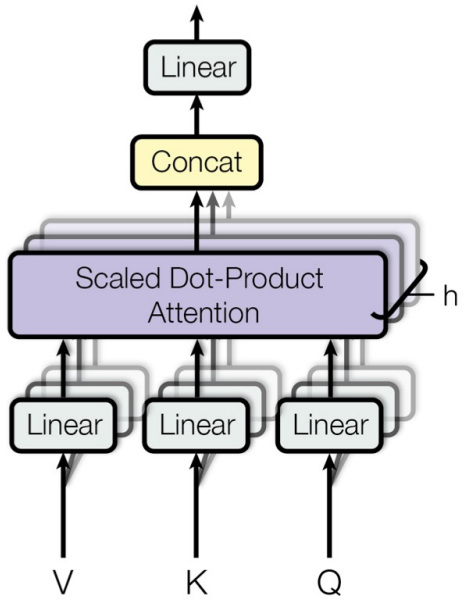


Image Source: Vaswani et al., 2017

Why "Multi-Head"?

- Multihead Attention implements *N* heads:
  - Split each projected observation vector into *N* parts
  - Each head is responsible for timeseries comprising of the corresponding split of the observations

Data Preprocessing

Data Preprocessing comprises of multiple preliminary steps:

1. Pivoting (Transforming a “long” table of event Sequences to a wide format)
2. Normalization
3. Resampling (Subdividing each timeseries into time bins of equal time length)
    1. Used aggregation method: averaging the observations in each time bin
4. Keeping track of the missingness of each observation
5. Forward and backward imputation of the missing values

$x_{1,1}$	0	0	$x_{1,4}$	$\cdots$	$x_{1,n_{ts}}$
1	0	0	1	$\cdots$	1

$x_{2,1}$	$x_{2,2}$	0	0	$\cdots$	$x_{2,n_{ts}}$
1	1	0	0	$\cdots$	1

0	0	$x_{3,3}$	$x_{3,4}$	$\cdots$	0
0	0	1	1	$\cdots$	0

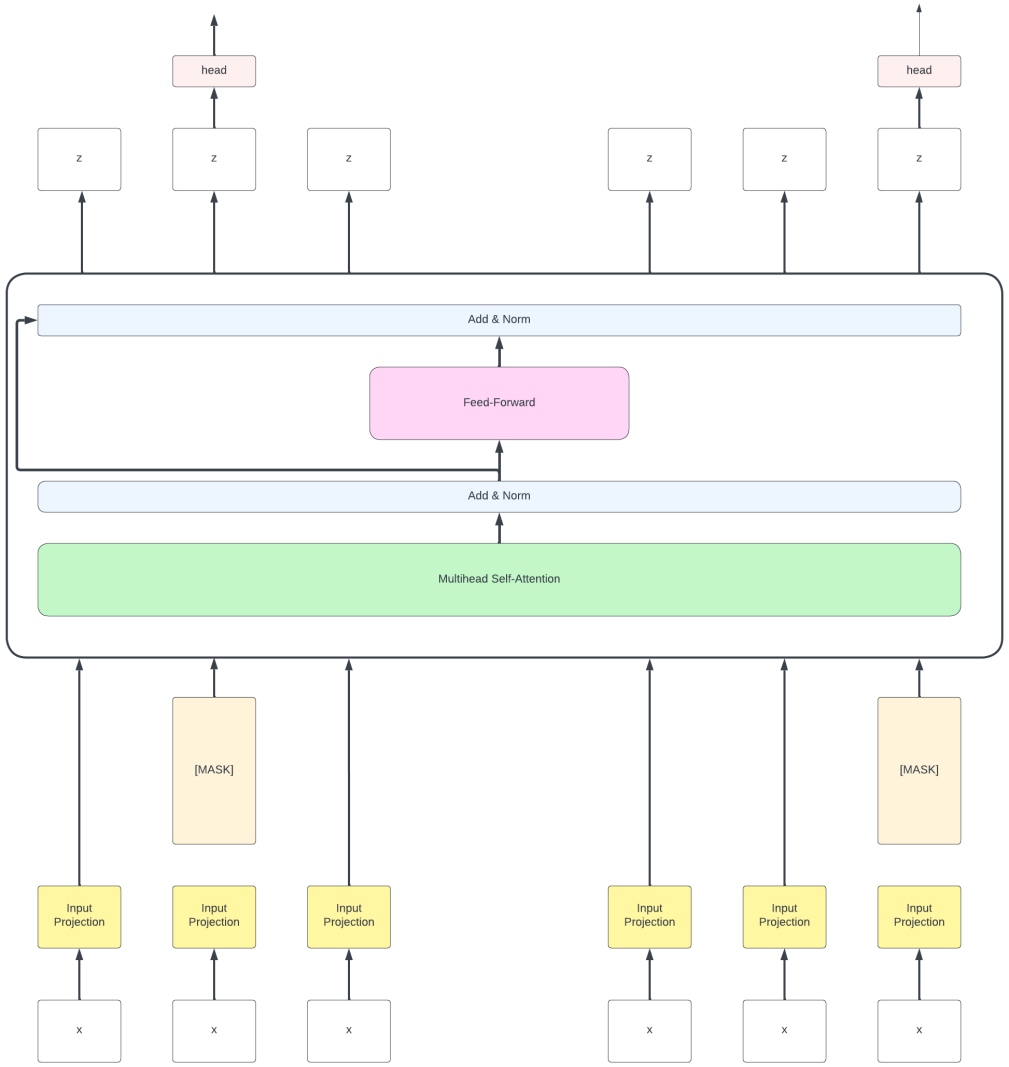
$\vdots$

$x_{L,1}$	0	$x_{L,3}$	0	$\cdots$	$x_{L,n_{ts}}$
1	0	1	0	$\cdots$	1

Self-Supervised Training

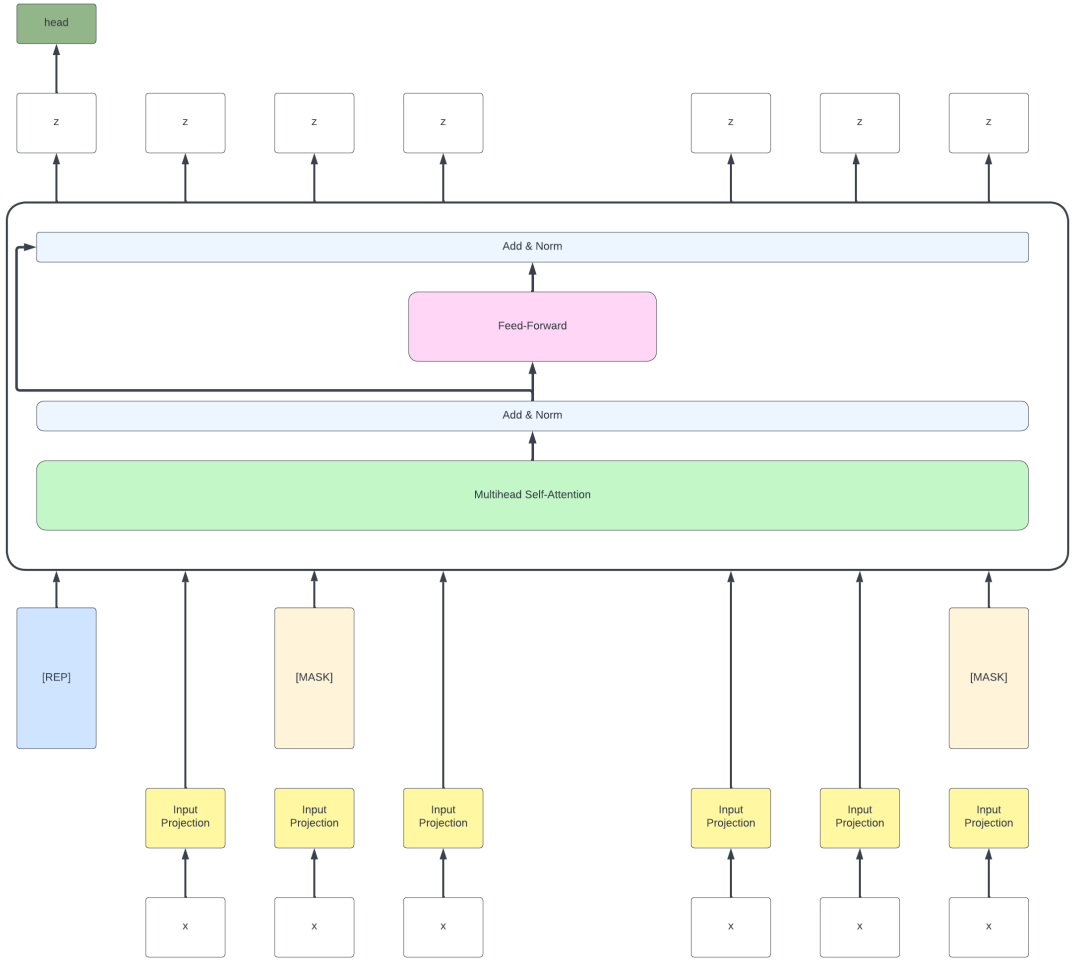
- Is performed by masked modeling approach:
  - Time bins are masked with a given probability
  - The training objective is to reconstruct observed values and the observed status of the variables in a time bin
  - Formally, we minimize average reconstruction loss per bin

$$\mathcal{L}_{\text{bin}} = \frac{1}{d}\sum_{i=1}^{d} m_{l,i}(\hat{x}_{l,i} - x_{l,i})^2 - \alpha(m_{l,i}\log(\hat{m}_{l,i}) + (1 - m_{l,i})\log(1 - \hat{m}_{l,i}))$$



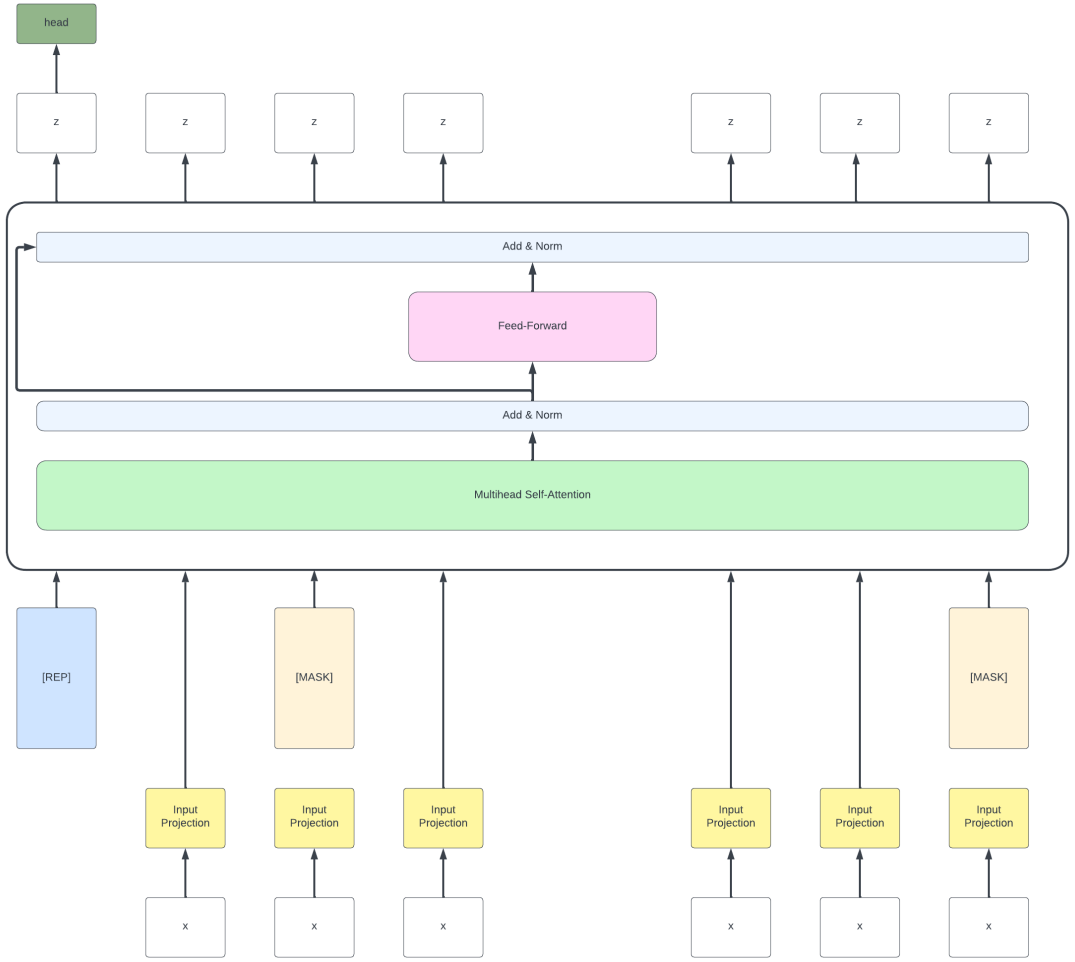
Finetuning

- the prior knowledge obtained from the pretraining stage can be used for the downstream task
- Classification head is applied to the encoded [REP] token
- The objective is chosen to be minimizing the cross-entropy loss



Finetuning

- Finetuning is usually performed on a smaller dataset, where the prior knowledge obtained from the pretraining stage can be used for the downstream task
- Classification head is applied to the encoded [REP] token
- The objective is chosen to be minimizing the cross-entropy loss



Special Tokens

- Tokenization is the process of projecting a time bin to a representation, which can be well handled by the encoder
- Special tokens:
  - [MASK]:
    - Lets the model learn dependencies between time bins during the pretraining
    - Is empirically found to be a regularization method during finetuning
  - [REP]:
    - Learns representation of the whole time sequence

Optimization methods: Introduction

- Both for Pretraining and Finetuning we optimize loss over the parameters of the model
- The classical optimization approach for Deep Learning is the Backpropagation algorithm, where a version of gradient descent is performed.
- The simplest version suited for large datasets is Stochastic Gradient Descent.

$$w := w - \eta \nabla Q_i(w).$$

- Due to noisyness, SGD converges slowly.

Decoupling Weight Decay from gradient-based Update (AdamW)

Algorithm 1 AdamW

repeat

$t \leftarrow t + 1$

$\nabla f_t(\theta_{t-1}) \leftarrow \text{SelectBatch}(\theta_{t-1})$

$g_t \leftarrow \nabla f_t(\theta_{t-1})$

$m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t$

$v_t \leftarrow \beta_2 m_{t-1} + (1 - \beta_1) g_t^2$

$\hat{m}_t \leftarrow \frac{m_t}{1-\beta_1^t}$

$\hat{v}_t \leftarrow \frac{v_t}{1-\beta_2^t}$

$\eta_t \leftarrow \text{SetScheduleMultiplier}(t)$

$\theta_t \leftarrow \theta_{t-1} - \eta_t (\alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda \theta_{t-1})$

until requirement met

return θ

Learning Rate Schedulers

- Linear Warmup
  - Linearly increase the learning rate until a fixed value is reached. Then decrease exponentially

- Cosine Annealing

$$\eta_t = \eta_{min}^i + \frac{1}{2}\left(\eta_{max}^i - \eta_{min}^i\right)\left(1 + \cos\left(\frac{T_{cur}}{T_i}\pi\right)\right)$$

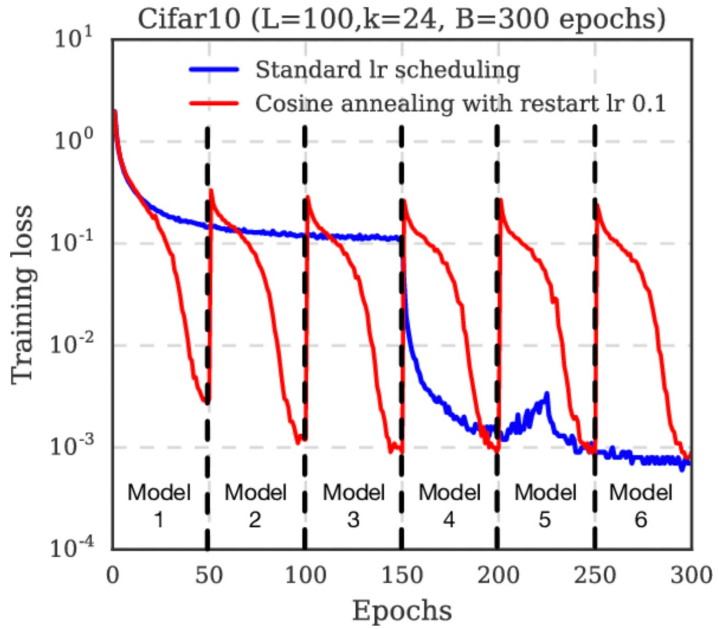
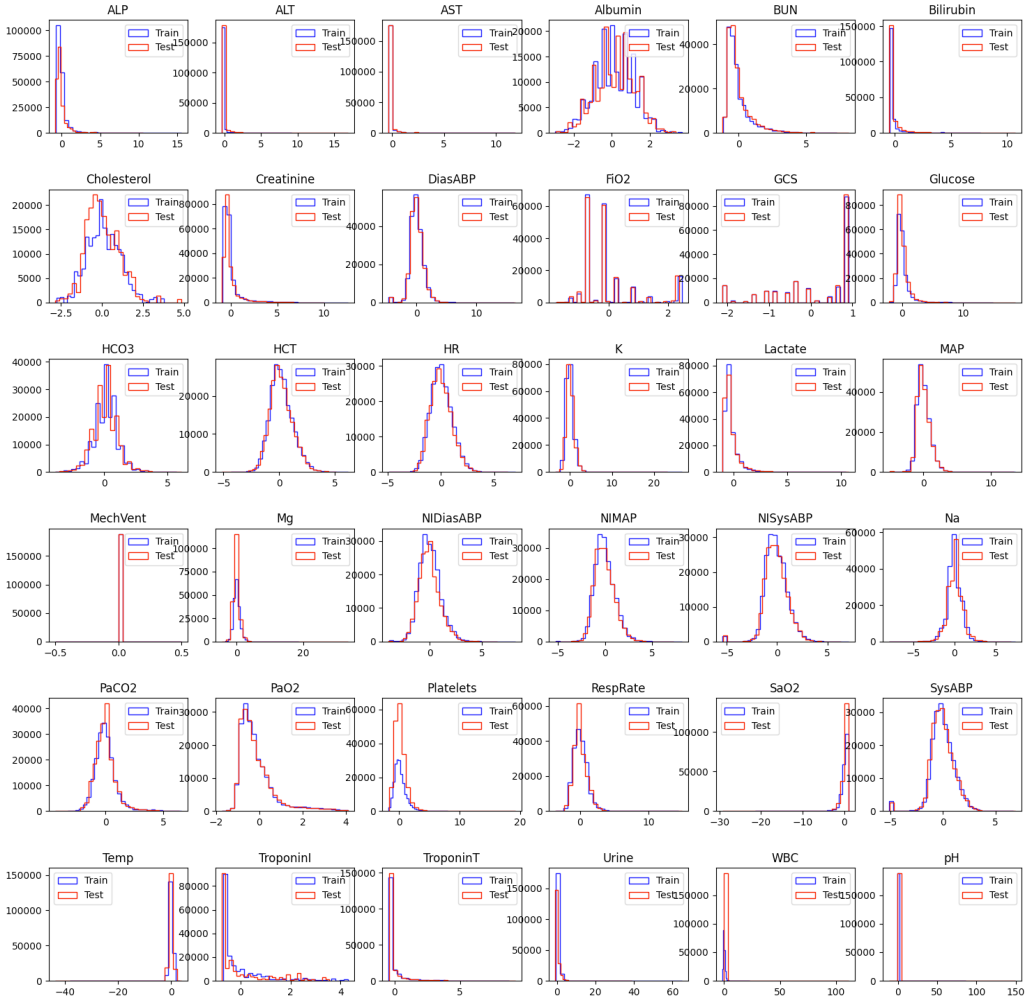


Image Source: Research Gate Gao Huang

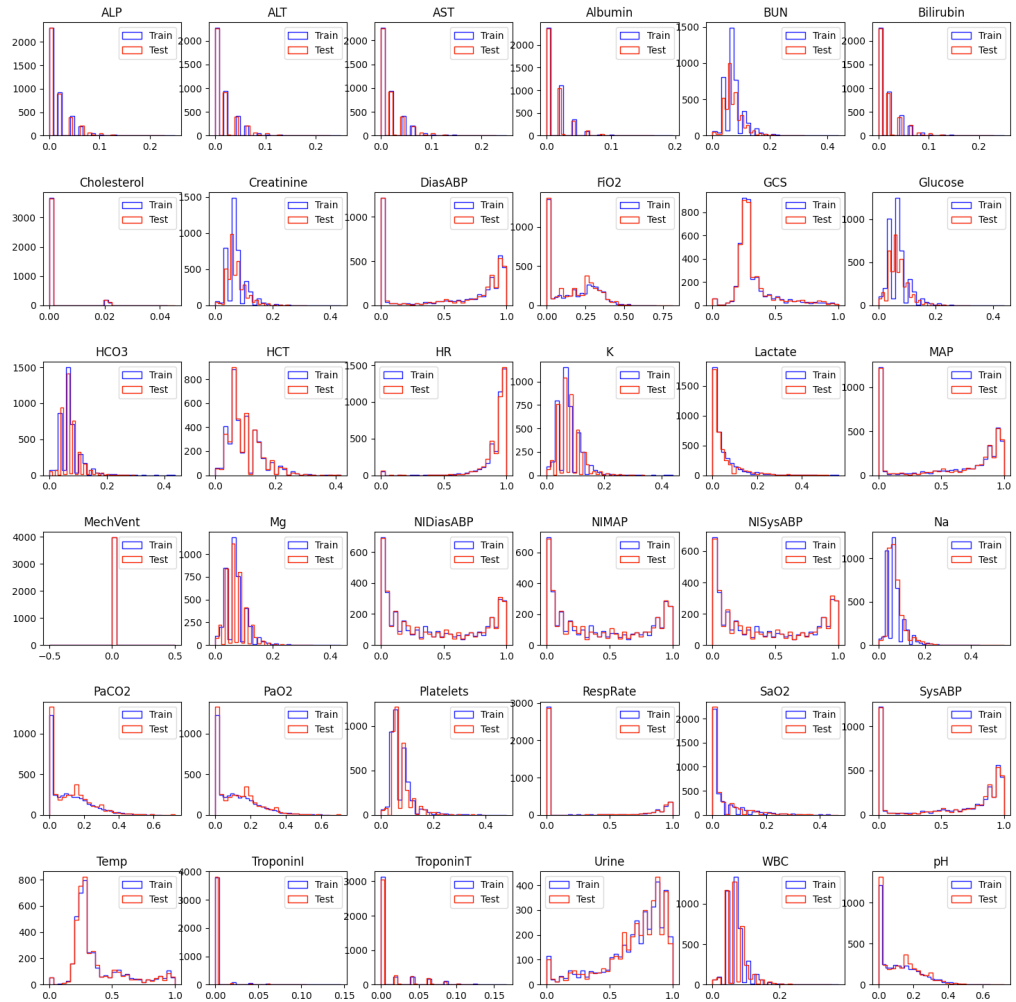
Experiments

- Physionet 2012 challenge
- EHR Data from 8K ICU stays
- 38 Timeseries variables reflecting vital patient metrics
- 5 Static variables (basic demographic & physical attributes)
- Max. length 48 Hours
- Target variable: In-hospital death
- Train/Val Split: 4K/4K

Dataset distribution



Distribution of Observations



Distribution of missingness

Results

	Self-Supervised Pretraining	Supervised Training
Hyperparameters	**Optimizer**: AdamW α: 0.2 **Masking prbability**: 0.2 **Dropout**: 0.2 **Epochs**: max 350	**Optimizer**: AdamW **Masking probability**: 0.5 **Dropout**: 0.5 **Epochs**: max 600

bin width	Scheduler	AU-ROC	Training BCE	Validation BCE
1H	Linear Warm Up	0.83	0.77	0.98
	Cosine Annealing	0.81	0.66	1.2
2H	Linear Warm Up	0.85	0.74	0.97
	Cosine Annealing	0.83	0.65	1.1

Conclusion

- Presented an approach for applying transformers to sparse time series
  - As Transformers expect discrete regular input, binning is a crucial preliminary step
  - The represenation is learned during the self-supervised training stage and finetuned for the downstream classification task
- Proof of Concept on Physionet2012 Dataset:
  - High sparsity, missbalanced labels
  - Relatively short time series after binning
  - Small dataset
  - Model achieves good experimental results (Still results given by the paper are better)
- Limitations and possible improvements:
  - More effective integration of global features possible
  - Masking could also be combined with positional encodings
  - Transformers Could be combined with other deep learning approches
    - GP-VAE uses Variational Autoencoders with Gaussian Process Prior to learn prior distribution of partially observed timeseries
  - Discussed metrics are possible not optimal for tasks with missbalanced classes

References

- Labach, A., Pokhrel, A., Yi, S. E., Zuberi, S., Volkovs, M., & Krishnan, R. G. (2023). Effective Self-Supervised Transformers For Sparse Time Series Data. Retrieved from https://openreview.net/forum?id=HUCgU5EQluN
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention Is All You Need. arXiv preprint. Retrieved from https://arxiv.org/abs/1706.03762
- Loshchilov, I., & Hutter, F. (2019). Decoupled Weight Decay Regularization. arXiv preprint. Retrieved from https://arxiv.org/abs/1711.05101