Transformer based multi-horizon temporal modeling of heterogenous signals

B.TECH Project report submitted to

Indian Institute of Technology Kharagpur

in partial fulfillment for the award of the degree of

Bachelor of Technology

in

Electronics and Electrical Communication Engineering

by

Harshavardhan Alimi

(18EC10021)

Under the supervision of

Prof. Sourangshu Bhattacharya

Department of Computer Engineering

Indian Institute of Technology Kharagpur

Autumn Semester, 2021-22

November 23, 2021,

DECLARATION

I certify that

(a) The work contained in this report has been done by me under the guidance of my supervisor.

(b) The work has not been submitted to any other Institute for any degree or diploma.

(c) I have conformed to the norms and guidelines given in the Ethical Code of Conduct of the Institute.

(d) Whenever I have used materials (data, theoretical analysis, figures, and text) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references. Further, I have taken permission from the copyright owners of the sources, whenever necessary.

Date: November 23, 2021, (Harshavardhan Alimi)
Place: Kharagpur (18EC10021)

DEPARTMENT OF COMPUTER ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KHARAGPUR

KHARAGPUR - 721302, INDIA

CERTIFICATE

This is to certify that the project report entitled "**Transformer based multi-horizon temporal modeling of heterogenous signals**" submitted by **Harshavardhan Alimi** (Roll No. 18EC10021) to Indian Institute of Technology Kharagpur towards partial fulfillment of requirements for the award of the degree of Bachelor of Technology in Electronics and Electrical Communication Engineering is a record of bonafide work carried out by him under my supervision and guidance during Autumn Semester, 2021-22.

Date: November 23, 2021,
Place: Kharagpur

Prof. Sourangshu Bhattacharya
Department of Computer Engineering
Indian Institute of Technology Kharagpur
Kharagpur - 721302, India

Abstract

Name of the student: **Harshavardhan Alimi** Roll No: **18EC10021**

Degree for which submitted: **Bachelor of Technology**

Department: **Department of Computer Engineering**

Thesis title: **Transformer based multi-horizon temporal modeling of heterogenous signals**

Thesis supervisor: **Prof. Sourangshu Bhattacharya**

Month and year of thesis submission: **November 23, 2021,**

Time series data is prevalent among various domains such as financial markets, environmental science, transportation, and healthcare. External factors often influence time-series in the form of asynchronous events, which may lead to repercussions in time-series forecasting; however, this data often exhibits long-term and short-term temporal dependencies. There exists a model which jointly handles heterogeneous temporal sequences which consist of time-series and event sequences. But, the model uses recurrent neural networks for handling event sequences, which is inefficient when the event sequences are lengthy and take a significant amount of time as it follows sequential architecture. To address these issues, I proposed a model which uses transformer architecture for handling event sequences, It takes advantage of the self-attention mechanism to capture long-term dependencies while still being computationally efficient. In this work, I compare the existing model with our approach.

Contents

Chapter 1

Introduction

In our daily lives, we see temporal data all the time. Temporal data can be available in various types. Time-series data is a major type,(also called synchronous sequence e.g.: Weather conditions, Health monitoring, stock exchange, footfall)the data points are taken at evenly spread discrete time points, For example in Weather conditions, The temperature data at a place can be taken at equal intervals of time, we can use this data to forecast the future data points which can be help determine necessary actions to be taken. Event-sequence data is another kind of temporal data, (also known as asynchronous sequence e.g.: extreme weather conditions, interactions at social media, medical records) the data points are unevenly dispersed in continuous time-domain. Both types of data provide a plethora of information on the evolution of complex systems, and accurate forecasts are crucial for subsequent decision-making, especially in time-sensitive situations. Both types of temporal data may influence each other in some areas, for example, extreme weather conditions at a particular time will influence the Temperature time-series data at the same time.

Research gaps Both types of temporal data can be used to model the problem of forecasting long-term multi-horizon future time-series, By elevating the rich information of mutual influence between both datasets can yield accurate future forecasts than using models with single temporal data. There has been recent work on modeling heterogeneous temporal sequences for multi-horizon probabilistic forecasting, which uses event-sequence data along with Time-series data to forecast future time-series (Li et al., 2021b) However, This model uses an RNN-based model (Du et al., 2016) for processing event sequence, There are few significant drawbacks with RNN-based models. First, The models are unlikely to capture long-term dependencies over long lengths of sequences(even though they are equipped with forget gates and memory gates, e.g.: LSTM (Hochreiter and Schmidhuber, 1997) and GRU (Chung et al., 2014)). Second, The trainability of RNN based deep neural network models because of existing gradient explosion and gradient vanishing problems. Third, RNN's are modeled sequentially instead of parallel, This limits the inability to scale to large problems.

Objective and scope To address these issues I came up with an approach of modeling event sequences with Transformers (Zuo et al., 2020) instead of RNN-based models (Du et al., 2016), (Mei and Eisner, 2017)(This can also be extended for Time-series). The non-recurrent structure of Transformers (Zuo et al., 2020) facilitates efficient training of multi-layer models. Transformers based architecture use self-attention mechanisms to leverage long term dependencies in a long sequence, This model discards the problem of gradient descent and gradient explosion, When calculating dependencies across all events, Transformers supports full parallelism, which means that the computation between any two event pairs is independent of each other. This results in a model with high efficiency.

I experimented with the proposed model on a multivariate dataset which was used in the paper(Li et al., 2021b). The Model showed better results in some scenarios when compared with the existing models, Further error analysis has been done to check the effect of the long sequence during evaluation, and the proposed model exhibits better performance for long event sequences when compared with the existing model(Li et al., 2021b).

Chapter 2

Background

In this section, I briefly review time-series forecasting models, Variational Auto-encoder (Kingma and Welling, 2013), Variational Synergetic Multi-Horizon Network (Li et al., 2021b), Transformer Hawkes Process (Zuo et al., 2020)

2.1 Time-series forecasting

Traditional statistical forecast models for time-series are based on linear Auto regressive approaches such as Exponential Smoothing (Hyndman et al., 2008), ARIMA (Box et al., 2015), VAR (Lütkepohl, 2005), These models gives the best results in some situations and they were still competitive with upcoming forecasting models. To learn multiple time-series forecasting Deep neural networks architecture was introduced by fusing existing traditional statistical models, such as the Deep factor model (Helmut, 2005) and DeepAR (Salinas et al., 2020) The above-mentioned methods are applicable for even spaced samples i.e., time-series data.

2.2 Variational Auto-encoder (VAE)

Variational Auto-encoder (Kingma and Welling, 2013) learns to generate the new data through learning the distribution of latent variables of the input data, It is generative model which handles the latent variables efficiently in deep neural networks. This will learn the marginal probability $p_\theta(\mathbf{x})$ from the data $\mathbf{x}$. The following is the process of VAEs generating process: The prior distribution $p_\theta(\mathbf{z})$ generates the latent variable $\mathbf{z}$, $\mathbf{x}$ is generated from the generative model $p_\theta(\mathbf{x}|\mathbf{z})$ conditioned on $\mathbf{z}$. A recognition model $q_\phi(\mathbf{z}|\mathbf{x})$ is introduced as an approximate model for the true posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$ (as the computation of this distribution is complicated). Further, the conditional variational auto-encoder (Sohn et al., 2015) which extends the VAE to model conditional generative process $p_\theta(\mathbf{y}|\mathbf{x})$ of target $\mathbf{y}$ given features $\mathbf{x}$.

2.3 Variational Synergetic Multi-Horizon Network (VSMHN)

VSMHN (Li et al., 2021b) is a novel deep conditional generative model which jointly models the event sequence and time-series for learning the complex correlations between heterogeneous temporal sequences, To combine advances in deep point processes models and variational recurrent neural networks, a tailored encoder is developed. Furthermore, for batched training on unaligned sequences an auxiliary transition scheme and an aligned time coding are carefully designed. This model is based on conditional variational auto-encoder (Sohn et al., 2015), this can be effectively trained using stochastic variational inference. The joint representations of heterogeneous sequences are generated using Time-Aware Hybrid RNN, which will consider those events lies in the time range of considered time-series sequence(input) and uses RNN to model time-series sequence(which transits at a predetermined time) and for asynchronous events, RNN is modeled motivated by (Du et al., 2016)(which transits at each event) a shared time feature is used for both the models to learn them in the same time-line, These joint representations were used to generate the distribution of the latent variables which completes the encoder part in the CVAE(Sohn et al., 2015). The decoder part is designed using a stochastic RNN generation model, the generated data from this model will be assumed to follow a parametric distribution, the probabilistic predictions are generated using Monte-Carlo simulation.

2.4 Transformer Hawkes Process (THP)

The better results of modeling event sequences have been achieved using the concept of the temporal point process, which are a powerful tool for handling continuous-time event sequence such as, Recurrent Point Process(Xiao et al., 2017), RMTPP(Du et al., 2016), Neural Hawkes process (Mei and Eisner, 2017) all the models are based on recurrent neural networks. THP (Zuo et al., 2020) addresses the drawback in Recurrent neural network-based models by using the self-attention mechanism in multi-layers for assimilating both short-term and long-term dependencies in the model and enjoys the computational efficiency by using parallelism for event sequences(instead of sequential approach as in RNN-based models). self-attention module plays a key role in the THP model, temporal information is passed into the model which can be encoded analogous to the original positional encoding method (Vaswani et al., 2017) as temporal encoding, event embedding was generated for events, both the features will be passed into multi-head multi-layer(This multi-layer will capture the high-level dependencies) attention to get attention output, which is then fed into a positive-wise feed-forward neural network to obtain hidden representation,which can help determine the next event time and event type.

Based on the foregoing observations, if we model the THP (Zuo et al., 2020) into the VSMHN (Li et al., 2021b) for better handling of event sequences to improve the long-term dependencies and make the model computationally efficient by utilizing the parallelism in the THP.In conclusion, The model is mostly welcomed to improve the correlation achieved between the heterogeneous temporal sequences.

Chapter 3

Proposed Model

3.1 Problem Formulation

we consider a M-dimensional multivariate time-series data which are sampled uniformly, $\mathcal{X}_{1:T} = \{x_1, ..., x_{T-1}, x_T\}$ where $x_t \in \mathrm{R}^M$, is the Time series data till time T.let $(c_1, t_1), ..., (c_n, t_n)$ be the event type(c_i) and time(t_i) correspondingly of the unevenly sampled event-sequence and $1 < t_1 < t_2 < .... < t_n$ and $t_n \leq T$ i.e, all the events occurring in the duration of considered time-series,let $\mathcal{H}_t = \{(c_i, t_i)|t_i < T\}$.Goal of the model is to estimate the distribution of future time-series until τ points i.e., to predict distribution of $\mathcal{X}_{T+1:T+\tau}$ given the past history of time-series and event sequences:

$$p(\mathcal{X}_{T+1:T+\tau}|\mathcal{X}_{1:T}, \mathcal{H}_t) \tag{3.1}$$

3.2 Model Architecture

Given the past temporal data $\mathcal{P} = \{\mathcal{X}_{1:\mathcal{T}}, \mathcal{H}_t\}$ our motive is to model the probability distribution of the future time-series points $\mathcal{F} = \mathcal{X}_{T+1:T+\tau}$ a latent variable $\mathbf{Z}$ is introduced conditioned on the input data $\mathcal{P}$, a VAE conditioned on past data is trained to maximize the log-likelihood function of $\mathcal{F}$ given $\mathcal{P}$, which entails a difficult marginalization over the latent variable Z

$$p(\mathcal{F}|\mathcal{P}) = \int_Z p(\mathcal{F}, \mathbf{Z}|\mathcal{P}) d\mathbf{Z} = \int_Z p(\mathcal{F}|\mathbf{Z}, \mathcal{P}) p(\mathbf{Z}|\mathcal{P}) d\mathbf{Z} \tag{3.2}$$

The model can be trained by optimizing the ELBO (variational evidence lower bound) of the conditioned log-likelihood function $\mathcal{L}(\phi, \theta)$ shown as below:

$$\begin{aligned} logp_\theta &= KL((q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F}))||(p_\theta(\mathbf{Z}|\mathcal{P}, \mathcal{F}))) + \mathbb{E}_{q_\phi}[-logq_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F}) + logp_\theta(\mathbf{Z}, \mathcal{F}|\mathcal{P})] \\ &\geq -KL((q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F}))||(p_\theta(\mathbf{Z}|\mathcal{P}))) + \mathbb{E}_{q_\phi}[logp_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})] = \mathcal{L}(\phi, \theta) \end{aligned} \tag{3.3}$$

Here, recognition model $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$(also encoder part in CVAE) where ϕ being the parameters of the model is as approximate for the true posterior distribution $p_\theta(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ as the latter is difficult to model, The difference between the two distributions is evaluated by using the concept of **KL** divergence i.e., to make the two distributions as close as possible while training the model. The future distributions for the model are generated through the generative model i.e., decoder of CVAE is $p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})$, $p_\theta(\mathbf{Z}|\mathcal{P})$ being the prior distribution of latent variable $\mathbf{Z}$ conditioned on the past data $\mathcal{P}$.Then the motive is to parameterize the $\mathcal{L}$ with the parameters ϕ and θ KL term can be estimated easily by assuming the latent variable $\mathbf{Z}$ follows Gaussian Distribution The other expectation term in ELBO is not differentiable. Differentiable ELBO bound is obtained using Auto-Encoding Variational Bayes(Kingma and Welling, 2013).

3.3 Implementation

A temporal feature is shared among even and irregular sequences to jointly model them. Time aware hybrid model is introduced to jointly learn the representations of temporal heterogeneous sequences for prior model $p_\theta(\mathbf{Z}|\mathcal{P})$ and recognition model $q_\phi(\mathbf{Z}|\mathcal{P},\mathcal{F})$ as shown in figure 3.1

Time aware hybrid model if we consider $\mathcal{X}_{1:T}$ be the input time series data and $\mathcal{H}_t$ be the event sequence data then Firstly, time-series is transited through RNN(LSTM) for fixed time T and it is modeled as f_ϕ to get the hidden representation and the event input which contains temporal data, event type and also the difference between the occurrence of 2 events is modeled using (Zuo et al., 2020) as g_ϕ, An extra temporal feature extractor is given as input to Learn them all at the same time which includes hour-of-day, day-of-week, month-of-year and absolute time. The below shows the model for the hidden representations of both sequences.

$$\begin{aligned}
\mathbf{h}_t^{\mathcal{X}} &= f_\phi([\Phi(\mathbf{x}_t), \Psi(\mathbf{t}_t), \mathbf{h}_{t-1}{}^{\mathcal{X}}]) && for \quad t = 1, 2, ..T \\
\mathbf{h}_T^{\mathcal{H}} &= g_\phi([\mathbf{e}_{1:L}, \Psi(\mathbf{t}_{1:L})]) && L = length\ of\ events\ in\ sequence
\end{aligned} \tag{3.4}$$

where Φ is used as a feature extractor for multi-variable time-series data which is implemented using MLP. Ψ is used as a feature extractor for shared temporal features which is also implemented by MLP's.$\mathbf{e}_{1:L}$ represents the one-hot encoding of event types ($\mathbf{e}_j \in \mathbb{R}^K$), K is the number of types of events)and $\Psi(\mathbf{t}_{1:L})$, represents the temporal features for the L events.
Below shows the implementation of model g_ϕ using the Transformer:- This model has the input of one-hot encoded L events $\mathbf{e}_{1:L}$ and temporal features
$\Psi(\mathbf{t}_{1:L}) := (\mathbf{t}_{1:L})$ $(assume)$,

Firstly, temporal features are encoded to obtain Temporal embedding for each timestamp by using the below trigonometric function:

$$[\mathbf{z}(t_j)]_i = \begin{cases} cos(t_j/10000^{\frac{i-1}{M}}) & \text{if} \quad i \ is \ odd \\ cos(t_j/10000^{\frac{i-1}{M}}) & \text{if} \quad i \ is \ even \end{cases} \tag{3.5}$$

Here $\mathbf{z}(t_j)$ is a deterministic function, and $\mathbf{z}(t_j) \in \ \mathbb{R}^M$, M being the dimension of the embedding for the transformer.
Secondly, Event embedding is learned using the matrix (i.e., parameterized model) $\mathbf{U} \in \ \mathbb{R}^{M \ X \ K}$, K is the number of types of events, if $\mathbf{e}_j$ is a one-hot encoding of an event then $\mathbf{Ue}_j$ will be the corresponding event embedding, so, if we are given a past data $\mathcal{S} = (t_j, \mathbf{e}_j)_{j=1}^L(t_j \leq \mathbf{T})$

$$X = (UY + Z)^T \tag{3.6}$$

where Y = $[\mathbf{e}_1, \mathbf{e}_2, ..., \mathbf{e}_L]$, Y $\in \ \mathbb{R}^{k \ X \ L}$ represents collection of events one-hot encoding values,then UY becomes the event embedding for L events and Z = $[z(t_L), ..., z(t_L)]$, Z $\in \ \mathbb{R}^{M \ X \ L}$ is a collection of temporal encoded values of temporal features, temporal encoding and event embedding are summed to obtain X, observe X $\in \ \mathbb{R}^{L \ X \ M}$ where each row corresponds to the summation of both embeddings of each event. The embeddings are then passed through the self-attention module to obtain attention output **S** which is given by:

$$\mathbf{S} = Softmax(\frac{\mathbf{QK}^T}{\sqrt{M_K}})V, \quad \mathbf{K} = \mathbf{BZ}^K, \ \mathbf{Q} = \mathbf{XW}^Q, \ \mathbf{V} = \mathbf{XW}^V \tag{3.7}$$

where K, Q, V are key, query, value matrices respectively they are modeled using simple MLP. In practise, it is more helpful for data fitting to use multi-head self-attention to increase model flexibility. Distinct attention outputs $S_1, S_2, ..., S_H$ are computed using different sets of weights to help with this. Then final attention will look $\mathbf{S} = [S_1, S_2, ..., S_H]$ After that, the attention output $\mathbf{S}$ is sent into a position-wise feed-forward neural network, which generates hidden representations $\mathbf{h}$(t) of the input event sequence:

$$\mathbf{H} = MLP_\phi[MLP_\psi[S]] \quad \mathbf{h}(t_j) \;=\; \mathbf{H}[j,:] \tag{3.8}$$

The layers of multi-head attention and positive feed-forward layers are stacked over one upon each other where hidden representations from positive feed-forward are fed into the next multi-head attention block which is depicted in fig 3.1 with number 4X at the Transformer block, then the hidden representation matrix from the last layer $\mathbf{H}_{last}$is considered to give the event hidden representation $\mathbf{h}_T^{\mathcal{H}}$ for eq.3.4

$$\mathbf{h}_T^{\mathcal{H}} = \mathbf{H}_{last}[L,:] \tag{3.9}$$

Auxiliary Transition unit When modeling extremely lengthy sequences, it's common to divide time-series into batches that overlap at the time axis (Salinas et al., 2020). However, with the addition of event sequences, this method becomes problematic. Due to the irregular nature of events, multiple neighboring batches may share the same collection of events, with the same time, event type, and other characteristics. However, the prediction targets for neighboring batches can differ, making it difficult for the model to capture features inside the event sequence when the same event sequence input reflects distinct targets. Auxiliary Transition is used to tackle the problem. Its main idea is to let the asynchronous Transformer know the precise end time as the time-series RNN, which is done by making a 0 event type at the time T.As a result, we can train our model in batches without having to worry about data conflicts.

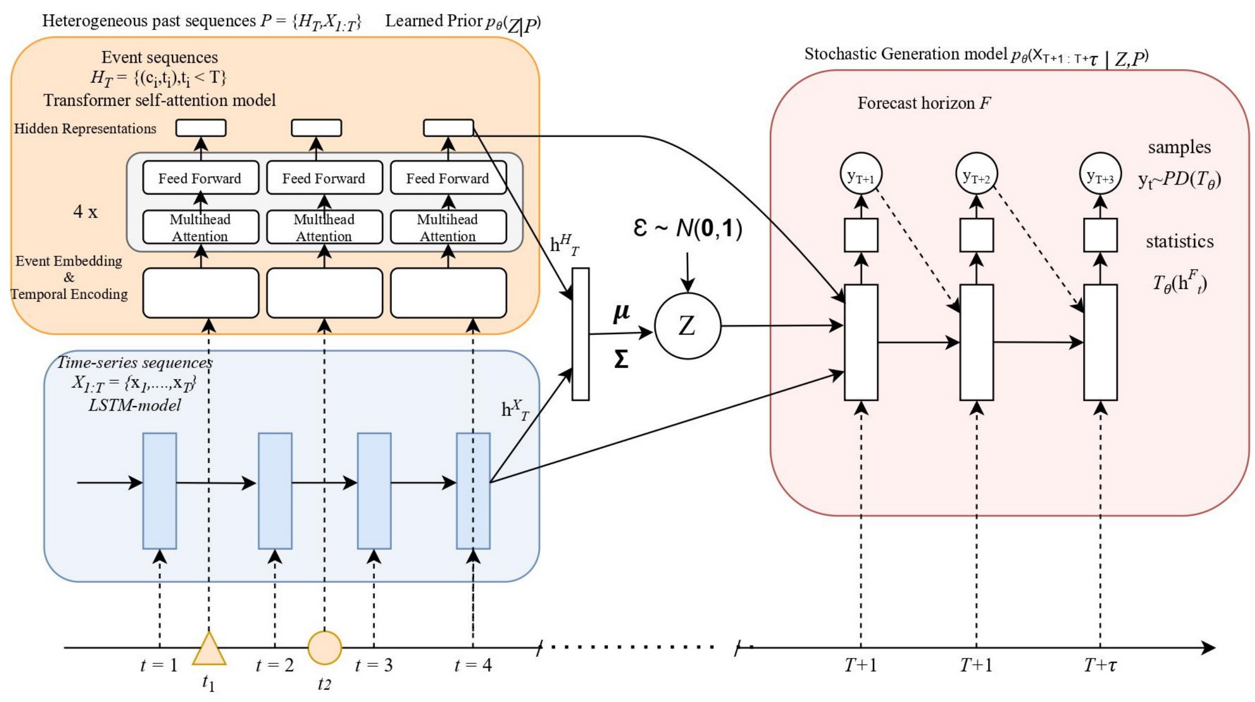
Heterogeneous past sequences $P = \{H_T, X_{1:T}\}$
Learned Prior $p_\theta(Z|P)$
Event sequences
$H_T = \{(c_i,t_i), t_i < T\}$
Transformer self-attention model
Hidden Representations
Feed Forward
Multihead Attention
4 x
Event Embedding & Temporal Encoding
Time-series sequences
$X_{1:T} = \{x_1,\dots,x_T\}$
LSTM-model
h^H_T
h^X_T
$\varepsilon \sim N(\mathbf{0},\mathbf{1})$
μ
Σ
Z
Stochastic Generation model $p_\theta(X_{T+1:T+\tau} | Z,P)$
Forecast horizon F
y_{T+1}
y_{T+2}
y_{T+3}
samples
$y_t \sim PD(T_\theta)$
statistics
$T_\theta(h^F_t)$
$t = 1$
t_1
$t = 2$
$t2$
$t = 3$
$t = 4$
T+1
T+1
$T+\tau$

FIGURE 3.1: our Proposed model for probabilistic forecasting of multi-horizon time series is presented in this overview. We set the history length M = 2, T = 4 event occurrences within the time range(and other event is auxilary transition unit), and the forecast horizon = 3 in the plot. The prior model $p_\theta(\mathbf{Z}|\mathcal{P})$ is shown on the left side. The recognition model (not displayed in the plot) employs the same time-series RNN parameters as the previous model during training. The recognition model transits τ more times with ground truth future value input before producing $h^{\mathcal{X}}_{T+\tau}$.It seeks to maximize the conditional probability $p(\mathcal{F}|\mathcal{P})$ of future value $\mathcal{F}$ given past $\mathcal{P}$ by minimising the sum of the KL-divergence between $q_\phi(\mathbf{Z}|\mathcal{P},\mathcal{F})$ and $p_\theta(\mathbf{Z}|\mathcal{P})$ as well as the negative log-likelihood of forecasted future time-series $p_\theta(\mathcal{X}_{T+1:T+\tau}|\mathbf{Z},\mathcal{P})$.

Synergetic layer We have obtained hidden representations $\mathbf{h}^{\mathcal{X}}_T$ and $\mathbf{h}^{\mathcal{H}}_T$ from temporal heterogeneous sequences using RNN and Transformer at time T, Now The difficulty then becomes parameterizing the prior model $p_\theta(\mathbf{Z}|\mathcal{P})$ and the recognition model $q_\phi(\mathbf{Z}|\mathcal{P},\mathcal{F})$. The latent variable is assumed to have a diagonal Gaussian distribution $Z \sim \mathcal{N}(\mu,\Sigma)$. The prior model $p_\theta(\mathbf{Z}|\mathcal{P})$ parameterization can be given as below:

$$\begin{aligned} \mu &= \varphi_\theta[\mathbf{h}_T^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}] \\ \Sigma &= diag(\varphi_\theta[\mathbf{h}_T^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}]) \end{aligned} \tag{3.10}$$

where φ is MLP with parameters θ,[a,b] denotes concatenation of features a,b. The recognition model $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ depends on past data $\mathcal{P}$ and also future data $\mathcal{F}$, this will be obtained when the time-series RNN tranits τ more times, The parameterization for the recognition model can be given below:

$$\begin{aligned} \mu &= \zeta_\phi[\mathbf{h}_{T+\tau}^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}] \\ \Sigma &= diag(\zeta_\phi[\mathbf{h}_{T+\tau}^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}]) \end{aligned} \tag{3.11}$$

where,ζ is MLP with parameters ϕ

```
Algorithm 1: Forecasting by Monte-Carlo sampling
  Input: Heterogeneous past data $\mathcal{P} = \{\mathcal{X}_{1:T}, \mathcal{H}_T\}$;
  Input: Trained model $p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})$ and $p_\theta(\mathbf{Z}|\mathcal{P})$;
  Input: Forecast horizon $\tau$ and number of samples $N$;
1 Evaluate Eq. 6 to get $\mathbf{h}_T^{\mathcal{X}}$ and $\mathbf{h}_T^{\mathcal{H}}$;
2 Compute $p_\theta(\mathbf{Z}|\mathcal{P}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by Eq. 8;
3 for $n = 1$ to $n = N$ do
4     Sample $\mathbf{Z} \sim \mathcal{N}(\mu, \Sigma)$;
5     $\mathbf{h}_T^{\mathcal{F}} = [\mathbf{h}_T^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}, \text{MLP}(Z)]$;
6     for $t = T + 1$ to $T + \tau$ do
7         $\mathbf{h}_t^F = d_\theta\left([\varphi(\mathbf{x}_{t-1}), \psi(\mathbf{t}_t)], \mathbf{h}_{t-1}^F\right)$;
8         $\mathbf{x}_t^{(n)} \sim \mathcal{PD}\left(\mathbf{T}_\theta(\mathbf{h}_t^F)\right)$;
9 Compute mean and quantiles from sampled $\mathbf{x}_t^{(n)}$,
  $n = 1 \ldots N$ for each time $t$;
```

FIGURE 3.2: Forecasting method: The Figure shows the algorithm used in the Monte-Carlo sampling for probabilistic forecasting of Multi-Horizon

Chapter 4

Experiments

To observe the persuasiveness of our model, The model is compared with the existing model (Li et al., 2021b). The error analysis has been done on the proposed model to see the positive effect of Transformer forecasting on long length sequences such that long-term dependencies are achieved. The dimensions of all hidden layers in RNN's and MLP's are kept at 100, the dimension of latent variable $\mathbf{z}$ is 50, 4 encoder layers are stacked upon each other in Transformer, 4 attentions are learned for each input to Transformer(which are after generates 1 hidden representation), 1000 trails of Monte-Carlo simulations are done for probabilistic forecasting of each future time-point, we use gaussian observation model as parametric distribution in stochastic RNN generation model, and adam optimiser is used with learning rate=0.001.

4.1 Dataset

The dataset is a multivariate dataset, 4 temporal features have been extracted as a shared feature between asynchronous events and Time-series, They are,hour-of-day, month-of-year, day-of-week, and absolute time for hourly sampled time-series data. 10% of Time-series Data(last 10% heterogeneous temporal data) is held for evaluation of the proposed model, The model is trained over the past week(168 data points) and future day(24 data points) and events held in the past week.

Environment This is a public air quality multi-variate dataset (Li et al., 2021a) consisting of Time-series data for: PM2.5, dew point, temperature, and pressure which are hourly sampled. The events considered are wind start, wind stop, and rain start which are extracted from the minute-level dataset.

4.2 Data preprocessing

The shingling method divides a lengthy time series into smaller parts. a batch of window width of W is considered at random start points [1, T - W + 1] from the dataset and input into the model at each training iteration. For event sequences, The inter-time difference is calculated $\triangle t$ as an event duration feature, and the event data is joined into time-series data into the same chunk, The events are padded at the start with zeros to maintain the same length of event sequence in a batch, which is highly followed in a Hawkes Process.4 Temporal features(shared feature): absolute time,hour-of-day, day-of-week, and month-of-year are extracted similar to positional encoding. To make the training easier the time-series Data is modified into $\mathcal{N}(0,1)$ distribution and similarly the Temporal features.

4.3 Compared Baselines

We have compared our proposed model implementation with the following 2 models.

VSMHN (Li et al., 2021b) jointly models the heterogeneous temporal data, The event sequences and Time-series are handled by the LSTM network, and it is also designed to handle the multivariate data.

VSMHN-TS slightly modifying the model (Li et al., 2021b)(which jointly models the heterogeneous temporal data) to consider only the Time-series, Events are not used in this model.

4.4 Evaluation Metric

The properscoring python module calculates CRPS analytically. The mean of the forecast distribution is used to determine the Root Mean Square Error (RMSE). All scores are based on the original data scale. The RMSE score will be the root mean square error of all time-series variables,i.e., $i = 1,\ldots, M$,(here in this dataset $M=4$) and over the future horizon,i.e., $t = T+1,\ T+2,\ldots,\ T+\tau$ it is given by:

$$RMSE = \sqrt{\frac{1}{M \times \tau} \times \sum_{i,t}(x_{i,t} - \widehat{x}_{i,t})^2} \tag{4.1}$$

where $x_{i,t}$ is the true future prediction for variable i at time point t and $\widehat{x}_{i,t}$ is the mean of the future distribution for variable i at time point t predicted by the model.

Gaussian observation model is used for computing CRPS score, If the predicted distribution for variable i follows distribution $\mathcal{N}(\mu, \sigma)$ then the CRPS score can be obtained as below:

$$crps[\mathcal{N}(\mu,\sigma), x] = \sigma\{\frac{x-\mu}{\sigma}[2\phi(\frac{x-\mu}{\sigma}) - 1] + 2\varphi(\frac{x-\mu}{\sigma}) - \frac{1}{\sqrt{\pi}}\} \tag{4.2}$$

where $\varphi(\frac{x-\mu}{\sigma})$ and $\phi(\frac{x-\mu}{\sigma})$ denotes CDF and PDF of the normal distribution with mean zero and variance one. For time-series with M variables can be averaged as below:

$$CRPS = \frac{1}{M}\sum_i crps(F_i, x_i) \tag{4.3}$$

4.5 Results

Method	PM2.5	Dewpoint	Temperature	Pressure
VSMHN-TS	67.92/32.21	3.51/1.81	2.69/1.26	2.58/1.43
VSMHN	66.73/33.33	**3.48**/1.85	**2.32/1.29**	**2.53/1.40**
Proposed model	**62.94/31.00**	3.54/**1.81**	2.43/1.35	2.55/**1.40**

Table 4.1: RMSE/CRPS scores comparison for the end 10% Time-series data(and events lying in that timeline)

Event sequence Range	PM2.5	Dewpoint	Temperature	Pressure
0-1 events	**86.72/42.18**	5.24/**2.67**	**2.21/1.20**	2.92/**1.59**
1-2 events	**59.56/30.12**	3.19/**1.67**	2.28/1.25	**2.25**/1.27
2-3 events	**58.39/29.10**	3.10/**1.64**	2.55/1.40	2.66/1.44
3-4 events	**51.03/25.40**	3.29/1.75	2.56/1.42	2.83/1.53
4-5 events	47.97/**25.09**	3.53/1.89	2.54/1.43	2.46/1.39
5-6 events	51.45/26.15	3.35/**1.72**	2.60/1.46	2.33/**1.28**
6-7 events	**58.35/29.92**	**2.77/1.44**	2.36/1.34	2.10/**1.18**
7-8 events	**65.91/34.28**	**2.34/1.27**	**2.04/1.17**	**2.25/1.41**
8-9 events	**77.86/39.94**	**2.49/1.33**	**1.89/1.07**	**2.25/1.39**
9-10 events	**110.90/65.54**	**3.10/1.76**	1.76/0.99	**1.89/0.98**

Table 4.2: RMSE/CRPS scores comparison in error analysis for proposed model

Event sequence Range	PM2.5	Dewpoint	Temperature	Pressure
0-1 events	95.93/48.70	**5.14**/2.82	2.35/1.30	**2.85**/1.60
1-2 events	68.73/34.91	**3.30**/1.79	**2.20/1.23**	2.26/**1.27**
2-3 events	64.04/31.90	**3.05**/1.66	**2.32/1.27**	**2.62/1.39**
3-4 events	53.6726.96	**3.10/1.68**	**2.37/1.30**	**2.72/1.44**
4-5 events	**47.51**/25.37	**3.35/1.79**	**2.40/1.34**	**2.42/1.37**
5-6 events	**49.68/26.02**	**3.33**/1.75	**2.48/1.41**	**2.30**/1.31
6-7 events	59.56/31.41	2.90/1.59	**2.31/1.33**	**2.06**/1.19
7-8 events	70.09/36.83	2.46/1.41	2.11/1.22	2.43/1.53
8-9 events	83.19/43.77	2.66/1.49	2.10/1.22	2.66/1.69
9-10 events	124.66/77.28	3.88/2.42	**1.60/0.93**	2.14/1.06

Table 4.3: RMSE/CRPS scores comparison in error analysis for VSMHN model

4.5.1 Performance comparison

The proposed model is compared with the baselines mentioned above.
Table 4.1(RMSE & CRPS scores) shows the proposed model out performs the state-of-art joint model in 1 variable out of 4 variables, which is because the validation set contains mostly of shorter length of event sequences(Transformer outperforms the RNN based models in the longer sequences)

4.5.2 Error analysis

To test the effect of the Transformer architecture on the joint model, the models were validated in the following way: Subsets of the validation set were chosen for validation based on the length of the event sequences (i.e, consider only those sequences which contain a particular range of events in event-sequence).
Tables 4.2 and 4.3 are the tables for the RMSE/CRPS scores of validation subsets as mentioned in the Description. The **BOLD** in the two tables for values describes the best value among corresponding values of two models at each validation subset and at each variable(comparison between two tables i.e., validation results of two models).
Table 4.2 shows the proposed model is giving the best results for the 1 variable over the entire subsets out of 4 variables, and all the variables are having the best results in the proposed model when the event sequence length is longer, as seen when the considered validation set is { 6-7, 7-8, 8-9, 9-10 events in event sequence }, This is to be expected, as the Transformer architecture takes precedence over all RNN-based models because it is meant to elevate long-term dependencies in the model that has learned thanks to the self-attention mechanism module.

Chapter 5

Future work and conclusion

5.1 Future work

In the proposed model the transformer architecture is used only for modeling asynchronous event sequences, but one can use transformer architecture to model the time-series. The work can be extended by implementing the proposed model on the dataset which contains the higher number of events in a considered heterogeneous(i.e., it should be in comparative size to the time-series sequence).

5.2 Conclusion

I conclude that in this paper a model is proposed which jointly handles the asynchronous and synchronous temporal data, where the model uses transformers self-attention mechanism to handle the asynchronous event sequences which leverage long-term relationships and is computationally efficient and the results show that the proposed model outperforms the baseline model VSMHN, when the event sequences given for validation contains a large number of events

Bibliography

Box, G. E., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*.

Du, N., Dai, H., Trivedi, R., Upadhyay, U., Gomez-Rodriguez, M., and Song, L. (2016). Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1555–1564.

Helmut, L. (2005). *New introduction to multiple time series analysis*. Springer Berlin Heidelberg.

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

Hyndman, R., Koehler, A. B., Ord, J. K., and Snyder, R. D. (2008). *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media.

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

Li, L., Yan, J., Yang, X., and Jin, Y. (2021a). Learning interpretable deep state space model for probabilistic time series forecasting. *arXiv preprint arXiv:2102.00397*.

Li, L., Zhang, J., Yan, J., Jin, Y., Zhang, Y., Duan, Y., and Tian, G. (2021b). Synergetic learning of heterogeneous temporal sequences for multi-horizon probabilistic forecasting. *arXiv preprint arXiv:2102.00431*.

Lütkepohl, H. (2005). *New introduction to multiple time series analysis*. Springer Science & Business Media.

Mei, H. and Eisner, J. (2017). The neural hawkes process: A neurally self-modulating multivariate point process. In *Proceedings of the 31st international conference on neural information processing systems*, pages 6757–6767.

Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. (2020). Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191.

Sohn, K., Lee, H., and Yan, X. (2015). Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Xiao, S., Yan, J., Yang, X., Zha, H., and Chu, S. (2017). Modeling the intensity function of point process via recurrent neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31.

Zuo, S., Jiang, H., Li, Z., Zhao, T., and Zha, H. (2020). Transformer hawkes process. In *International Conference on Machine Learning*, pages 11692–11702. PMLR.