EINN: Epidemic Prediction using ML

Mert Saruhan, B.Sc.

November 29, 2023

Agenda

What is EINN?

What is EINN?

Introduction

- EINN is a (**Neural Network**) machine learning model to predict epidemic dynamics [1]
- EINN combines the knowledge of PINN, RNN, and ODE [1]

What is EINN?

Introduction

- EINN is a (**Neural Network**) machine learning model to predict epidemic dynamics [1]
- EINN combines the knowledge of PINN, RNN, and ODE [1]

Background

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

SIR epidemic models

- Statical way to predict epidemic dynamics
- Uses ODE to predict
- Have many models: SIR, SEIR, SEIRM etc.
- Susceptible (S): $\frac{dS_t}{dt} = -\beta_t \frac{S_t I_t}{N}$
- Exposed (E): $\frac{dE_t}{dt} = \beta_t \frac{S_t I_t}{N} - \alpha_t E_t$
- Infected (I): $\frac{dI_t}{dt} = \alpha_t E_t - \gamma_t I_t - \mu_t I_t$
- Recovered (R): $\frac{dR_t}{dt} = \gamma_t I_t$
- Mortality (M): $\frac{dM_t}{dt} = \mu_t I_t$

Note: $\Omega_t = \{\alpha_t, \beta_t, \gamma_t, \mu_t\}$ are parameters, N is the population number, and t is time.

Background

RNN

- RNN is a neural network model
- Uses the previous state as an input to predict the next state
- Used in object detection and NLP

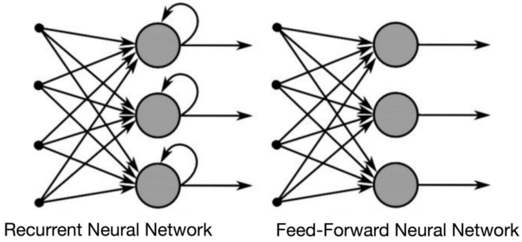

Figure: Example structure of RNN and NN [2]

Background

RNN

- RNN is a neural network model
- Uses the previous state as an input to predict the next state
- Used in object detection and NLP

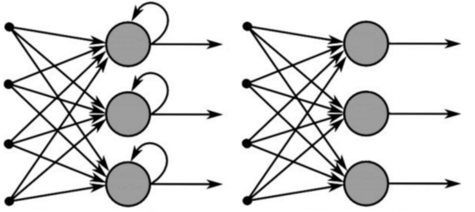

Figure: Example structure of RNN and NN [2]

Background

RNN

- RNN is a neural network model
- Uses the previous state as an input to predict the next state
- Used in object detection and NLP

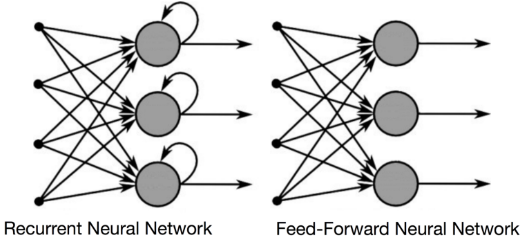

Figure: Example structure of RNN and NN [2]

Background

PINN

- Uses physics laws to predict the next state
- Uses PDE (partial differential equation) and ODE (ordinary differential equation) as a loss estimator
- $\mathrm{L}_{\mathrm{data}}$: loss from data fitting
- $\mathrm{L}_{\mathrm{physics}}$: loss from physics laws

  e.g. $\vec{a} = -\mu\|\vec{v}\|\vec{v} - \vec{g}, \vec{v} = \frac{df}{dt}, \vec{a} = \frac{d^2f}{dt^2}$

  $$\mathrm{L}_{\mathrm{physics}} = \frac{1}{N}\sum_{i=0}^{N-1}\left(-\mu\|\frac{df_i}{dt_i}\|\frac{df_i}{dt_i} - \vec{g} - \left(\frac{d^2f_i}{dt_i^2}\right)\right)^2$$
- $\mathrm{L}_{\mathrm{total}} = \mathrm{L}_{\mathrm{data}} + \mathrm{L}_{\mathrm{physics}}$

Note: $\vec{a}$ is acceleration, $\vec{v}$ is velocity, $\vec{g}$ is gravity, μ is friction, f is position, t is time, and N is the number of data points.

Background

PINN for Systems Biology

- Recent works with PINN for Systems Biology [3, 4] enables ODE for PINN
- Uses time t as input for Neural Network $N(t)$ and rate of change in ODE systems $f_{\text{ODE}}(t)$
- Uses $(\frac{dN(t)}{dt} - f_{\text{ODE}}(t))$ as a loss

HOCHSCHULE MITTWEIDA

EINN Model

EINN Model

Model summary

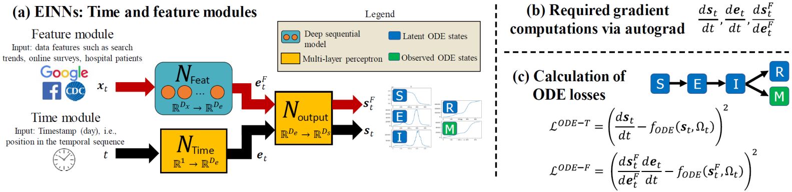

Figure: EINN Model. (a) The training model. (b) Needed gradients to train the model using autograd. (c) ODE losses of feature module and time module. Taken from: [1]

EINN Model

Source model (Time module)

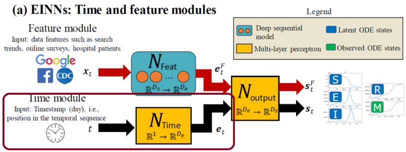

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Source model is a PINN model
- Input: time
- Output: embedding of the epidemic ODE states (e_t)
- Uses PINN for Systems Biology
- Model freezes when $e_t \approx e_t^F$

EINN Model

Source model (Time module)

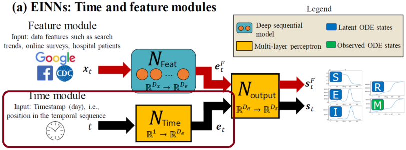

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Source model is a PINN model
- Input: time
- Output: embedding of the epidemic ODE states (e_t)
- Uses PINN for Systems Biology
- Model freezes when $e_t \approx e_t^F$

EINN Model

Source model (Time module)

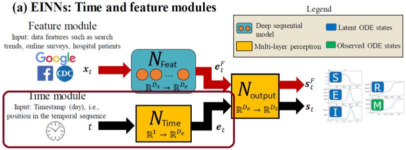

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Source model is a PINN model
- Input: time
- Output: embedding of the epidemic ODE states (e_t)
- Uses PINN for Systems Biology
- Model freezes when $e_t \approx e_t^F$

EINN Model

Source model (Time module)

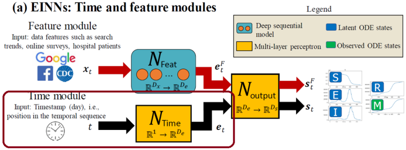

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Source model is a PINN model
- Input: time
- Output: embedding of the epidemic ODE states (e_t)
- Uses PINN for Systems Biology
- Model freezes when $e_t \approx e_t^F$

HOCHSCHULE MITTWEIDA

EINN Model

Time module losses

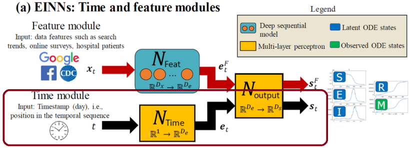

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- ODE loss for time module ($L^{\text{ODE}-T}$) [1]:
  $$\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[\frac{ds_t}{dt} - f_{\text{ODE}}(s_t, \Omega_t) \right]^2$$

EINN Model

Time module losses

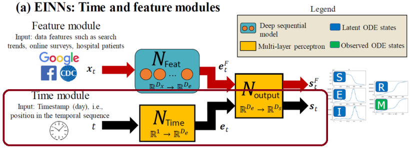

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- According to Wu et al. [5], in the ODE states, Mortality (M) is observed and other states are latent
- $L^{\text{Data-T}}$ [1]: $\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[\hat{M}_t - M_t\right]^2$

HOCHSCHULE MITTWEIDA

EINN Model

Time module losses

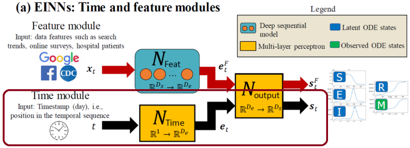

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- According to Wu et al. [5], in the ODE states, Mortality (M) is observed and other states are latent
- $L^{\text{Data-T}}$ [1]: $\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[\hat{M}_t - M_t\right]^2$

EINN Model

Time module losses

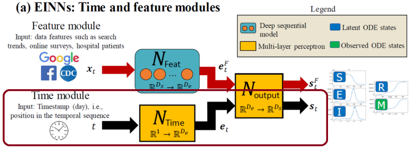

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Adding monoticity loss (loss from latent states) to make learning less challenging using domain knowledge
- $L^{\text{Mono}} = \frac{1}{N+1}\left(\sum_{t=t_0}^{t_N} \frac{dS_t}{dt} \text{ReLU}\left(\frac{dS_t}{dt}\right) + \sum_{t=t_0}^{t_N} -1 \frac{dR_t}{dt} \text{ReLU}\left(-\frac{dR_t}{dt}\right)\right)$

EINN Model

Time module losses

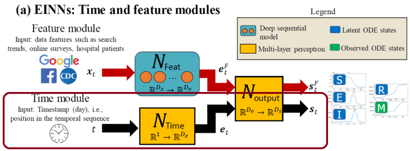

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Adding monoticity loss (loss from latent states) to make learning less challenging using domain knowledge
- $L^{\text{Mono}} = \frac{1}{N+1}\left(\sum_{t=t_0}^{t_N} \frac{dS_t}{dt}\text{ReLU}\left(\frac{dS_t}{dt}\right) + \sum_{t=t_0}^{t_N} -1\frac{dR_t}{dt}\text{ReLU}\left(-\frac{dR_t}{dt}\right)\right)$

HOCHSCHULE MITTWEIDA

EINN Model

Time module losses

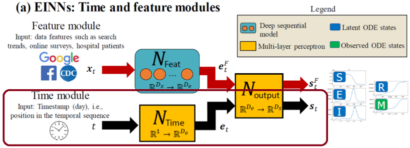

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Dealing time varying ODE model with parameter loss
- $L^{\text{Param}} = \frac{1}{N+1} \sum_{t=t_0}^{t_N} [\Omega_{t+1} - \Omega_t]^2$

EINN Model

Time module losses

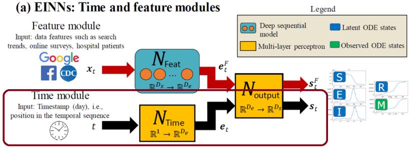

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Dealing time varying ODE model with parameter loss
- $L^{\text{Param}} = \frac{1}{N+1} \sum_{t=t_0}^{t_N} [\Omega_{t+1} - \Omega_t]^2$

HOCHSCHULE MITTWEIDA

EINN Model

Target model (Feature module)

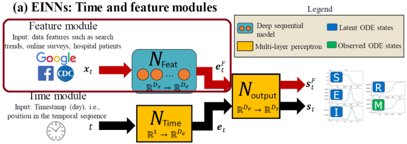

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Target model is an RNN model
- Input (x_t): Features relevant to the problem
- Output: embedding of the epidemic ODE states (e_t^F)
- Model freezes when $e_t \approx e_t^F$
- Loss (L^{Emb}): $\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[e_t - e_t^F \right]^2$

HOCHSCHULE MITTWEIDA

EINN Model

Target model (Feature module)

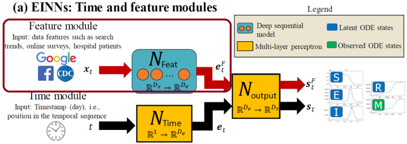

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Target model is an RNN model
- Input (x_t): Features relevant to the problem
- Output: embedding of the epidemic ODE states (e_t^F)
- Model freezes when $e_t \approx e_t^F$
- Loss (L^{Emb}): $\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[e_t - e_t^F\right]^2$

HOCHSCHULE MITTWEIDA

EINN Model

Target model (Feature module)

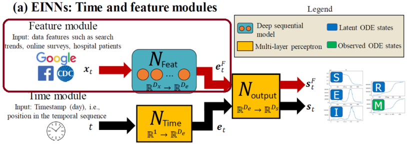

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Target model is an RNN model
- Input (x_t): Features relevant to the problem
- Output: embedding of the epidemic ODE states (e_t^F)
- Model freezes when $e_t \approx e_t^F$
- Loss (L^{Emb}): $\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[e_t - e_t^F\right]^2$

EINN Model

Target model (Feature module)

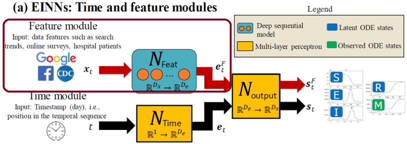

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Target model is an RNN model
- Input (x_t): Features relevant to the problem
- Output: embedding of the epidemic ODE states (e_t^F)
- Model freezes when $e_t \approx e_t^F$
- Loss (L^{Emb}): $\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[e_t - e_t^F\right]^2$

EINN Model

Target model (Feature module)

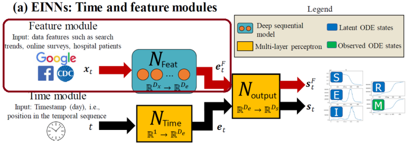

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Target model is an RNN model
- Input (x_t): Features relevant to the problem
- Output: embedding of the epidemic ODE states (e_t^F)
- Model freezes when $e_t \approx e_t^F$
- Loss (L^{Emb}): $\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[e_t - e_t^F\right]^2$

HOCHSCHULE MITTWEIDA

EINN Model

Target module losses

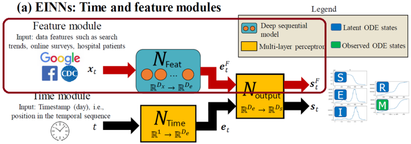

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $e_t \approx e_t^F$ required for gradient trick for ODE loss from feature module
- $e_t \approx e_t^F \Rightarrow \frac{ds_t^F}{dt} = \frac{ds_t^F}{de_t^F}\frac{de_t^F}{dt} \approx \frac{ds_t^F}{de_t^F}\frac{de_t}{dt}$
- Loss ODE from feature module ($L^{\text{ODE}-F}$) [1]:

  $\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[\frac{ds_t^F}{de_t^F}\frac{de_t}{dt} - f_{\text{ODE}}(s_t^F, \Omega_t)\right]^2$

EINN Model

Target module losses

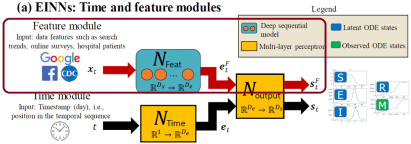

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $e_t \approx e_t^F$ required for gradient trick for ODE loss from feature module
- $e_t \approx e_t^F \Rightarrow \frac{ds_t^F}{dt} = \frac{ds_t^F}{de_t^F}\frac{de_t^F}{dt} \approx \frac{ds_t^F}{de_t^F}\frac{de_t}{dt}$
- Loss ODE from feature module (L^{ODE-F}) [1]:

$$\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[\frac{ds_t^F}{de_t^F}\frac{de_t}{dt} - f_{ODE}(s_t^F, \Omega_t)\right]^2$$

EINN Model

Target module losses

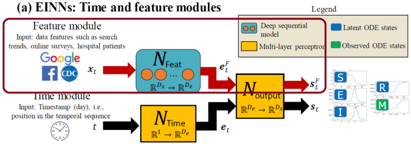

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $e_t \approx e_t^F$ required for gradient trick for ODE loss from feature module
- $e_t \approx e_t^F \Rightarrow \frac{ds_t^F}{dt} = \frac{ds_t^F}{de_t^F}\frac{de_t^F}{dt} \approx \frac{ds_t^F}{de_t^F}\frac{de_t}{dt}$
- Loss ODE from feature module ($L^{\text{ODE}-F}$) [1]:

$$\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[\frac{ds_t^F}{de_t^F}\frac{de_t}{dt} - f_{\text{ODE}}(s_t^F, \Omega_t)\right]^2$$

EINN Model

Target module losses

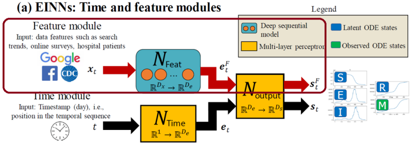

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $\hat{M}_t^F$: predicted mortality using feature module path
- Loss from data ($L^{\text{Data}-F}$):
  $\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[\hat{M}_t^F - M_t\right]^2$

HOCHSCHULE MITTWEIDA

EINN Model

Target module losses

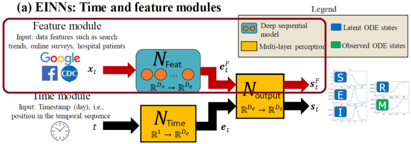

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $\hat{M}_t^F$: predicted mortality using feature module path
- Loss from data ($L^{\text{Data}-F}$): $\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[\hat{M}_t^F - M_t\right]^2$

EINN Model

Target module losses

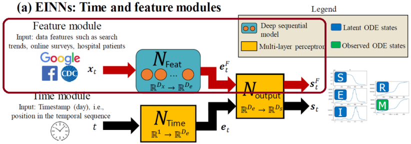

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Alligning findings from both paths
- $L^{\text{Output}} = \frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[s_t - s_t^F\right]^2$

EINN Model

Target module losses

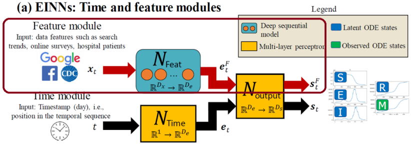

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Alligning findings from both paths
- $L^{\text{Output}} = \frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[s_t - s_t^F\right]^2$

HOCHSCHULE MITTWEIDA

Results

Results

Bibliography I

[1] A. Rodríguez, J. Cui, N. Ramakrishnan, B. Adhikari, and B. A. Prakash, *Einns: Epidemiologically-informed neural networks*, 2023. DOI: 10.48550/arXiv.2202.10446. arXiv: 2202.10446 [cs.LG].

[2] *A guide to recurrent neural networks: Understanding rnn and lstm networks*, https://builtin.com/data-science/recurrent-neural-networks-and-lstm, Accessed: 2023-11-26.

[3] A. Yazdani and et al., *Systems biology informed deep learning for inferring parameters and hidden dynamics*. PLOS Comp. Bio., 2020.

[4] G. Karniadakis and et al., *Physics-informed machine learning*, 2021.

Bibliography II

[5] J. T. Wu and et al., *Nowcasting and forecasting the potential domestic and international spread of the 2019-ncov outbreak originating in wuhan, china: A modelling study*, 2020.

HOCHSCHULE MITTWEIDA
University of Applied Sciences

Thank You

Mert Saruhan, B.Sc.

msaruhan@hs-mittweida.de

Mathematics for Network and Data Science (MA20w1-M)

Hochschule Mittweida
University of Applied Sciences
Technikumplatz 17 | 09648 Mittweida