EINN: Epidemic Prediction using ML

Mert Saruhan, B.Sc.

December 2, 2023

HOCHSCHULE MITTWEIDA
University of Applied Sciences

Agenda

1. What is EINN?
2. Background
3. EINN Model
4. Results

HOCHSCHULE MITTWEIDA

What is EINN?

What is EINN?

Introduction

- EINN is a (**Neural Network**) machine learning model to predict epidemic dynamics [1]
- Mechanistic models (ODE models e.g SIR/SEIRM) good at trend approximation [1]
- RNN powered epidemic models good at short and long term epidemic prediction [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]
- Mechanistic models (ODE models e.g SIR/SEIRM) good at trend approximation [1]
- RNN powered epidemic models good at short and long term epidemic prediction [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]
- Mechanistic models (ODE models e.g SIR/SEIRM) good at trend approximation [1]
- RNN powered epidemic models good at short and long term epidemic prediction [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]
- Mechanistic models (ODE models e.g SIR/SEIRM) good at trend approximation [1]
- RNN powered epidemic models good at short and long term epidemic prediction [1]
- EINN combines the knowledge of PINN, RNN, and ODE [1]

Background

Background

Mechanistic epidemic models

- 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

Mechanistic epidemic models

- 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

Mechanistic epidemic models

- 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

Mechanistic epidemic models

- 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

Mechanistic epidemic models

- 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.

HOCHSCHULE MITTWEIDA

Background

Mechanistic epidemic models

- 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

Mechanistic epidemic models

- 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

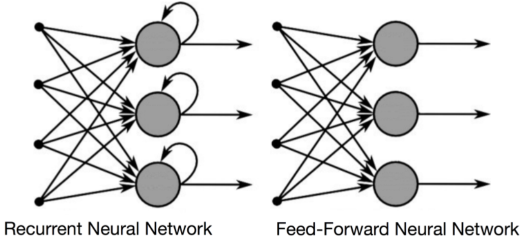

Figure: Example structure of RNN and NN [2]

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

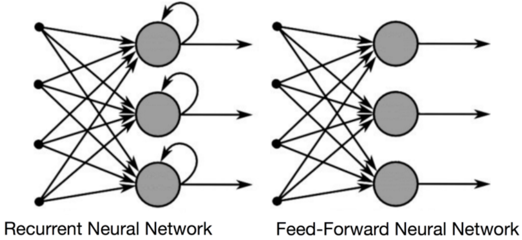

Figure: Example structure of RNN and NN [2]

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

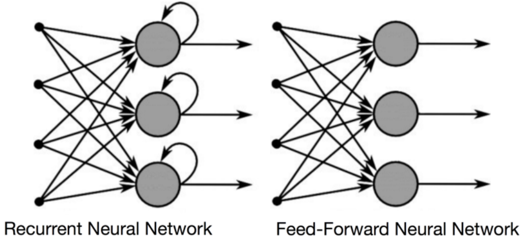

Figure: Example structure of RNN and NN [2]

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

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

- $\mathrm{L}_{\mathrm{total}} = \mathrm{L}_{\mathrm{data}} + \mathrm{L}_{\mathrm{physics}}$

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

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

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

  $$L_{physics} = \frac{1}{N}\sum_{i=0}^{N-1}\left(-\mu\left\|\frac{df_i}{dt_i}\right\|\frac{df_i}{dt_i} - \vec{g} - \left(\frac{d^2f_i}{dt_i^2}\right)\right)^2$$

- $L_{total} = L_{data} + L_{physics}$

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

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

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

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

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

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

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

  $$L_{\text{physics}} = \frac{1}{N} \sum_{i=0}^{N-1} \left(-\mu \| \frac{df_i}{dt_i} \| \frac{df_i}{dt_i} - \vec{g} - \left(\frac{d^2 f_i}{dt_i^2} \right) \right)^2$$

- $L_{\text{total}} = L_{\text{data}} + L_{\text{physics}}$

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

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

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

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

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]: better PINN for SIR mechanics
- Uses time t as input for Neural Network $N(t)$ and rate of change in ODE systems $f_{\mathrm{ODE}}(t)$
- Uses $(\frac{dN(t)}{dt} - f_{\mathrm{ODE}}(t))$ as a loss

Background

PINN for Systems Biology

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

Background

PINN for Systems Biology

- Recent works with PINN for Systems Biology [3, 4]: better PINN for SIR mechanics
- 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

EINN Model

EINN Model

Model summary

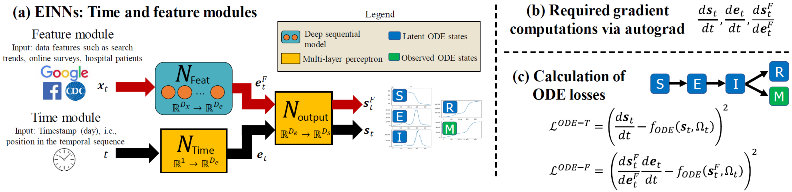

Figure: EINN Model. (a) The training model. (b) Needed gradients to train the model using autograd. (c) ODE losses of feature module and time module. Taken from: [1]

EINN Model

Source model (Time module)

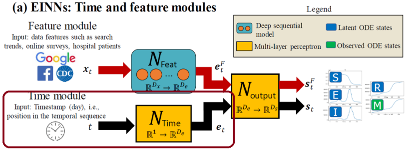

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

EINN Model

Source model (Time module)

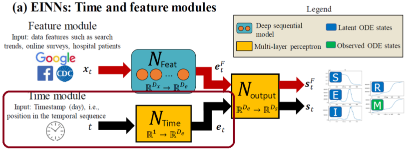

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

EINN Model

Source model (Time module)

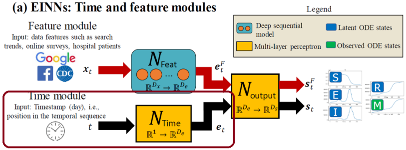

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

HOCHSCHULE MITTWEIDA

EINN Model

Source model (Time module)

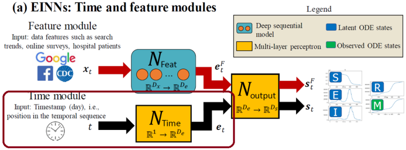

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

EINN Model

Source model (Time module)

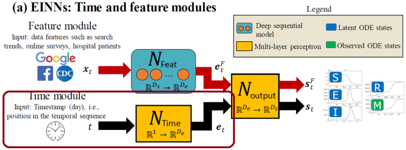

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

EINN Model

Time module losses

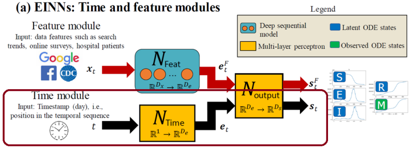

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

EINN Model

Time module losses

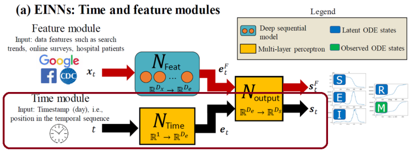

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

  M: mortality, $\hat{M}_t$: predicted mortality using time module path

EINN Model

Time module losses

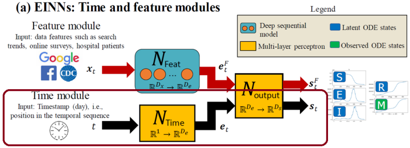

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- In the ODE states, Mortality (M) is observed and other states are latent [5]
- $L^{\text{Data-T}}$ [1]: $\frac{1}{N+1}\sum_{t=t_0}^{t_N}\left[\hat{M}_t - M_t\right]^2$
  M: mortality, $\hat{M}_t$: predicted mortality using time module path

EINN Model

Time module losses

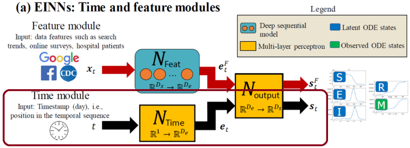

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

HOCHSCHULE MITTWEIDA

EINN Model

Time module losses

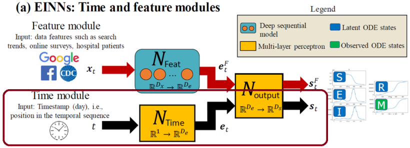

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

HOCHSCHULE MITTWEIDA

EINN Model

Time module losses

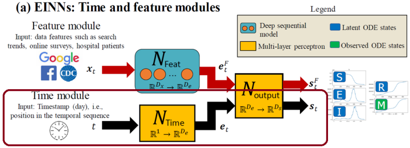

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Adding parameter to loss to track parameter change with time
- $L^{\text{Param}} = \frac{1}{N+1} \sum_{t=t_0}^{t_N} [\Omega_{t+1} - \Omega_t]^2$

EINN Model

Time module losses

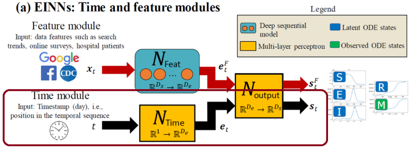

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- Adding parameter to loss to track parameter change with time
- $L^{\text{Param}} = \frac{1}{N+1} \sum_{t=t_0}^{t_N} [\Omega_{t+1} - \Omega_t]^2$

HOCHSCHULE MITTWEIDA

EINN Model

Target model (Feature module)

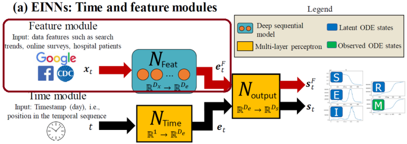

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- 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}): $\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[e_t - e_t^F\right]^2$

EINN Model

Target model (Feature module)

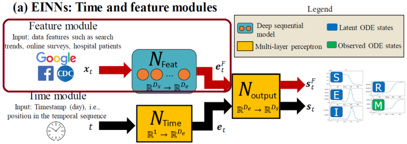

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- 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)

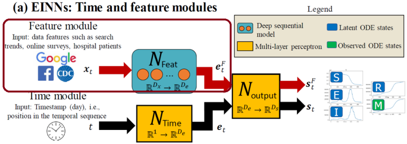

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- 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)

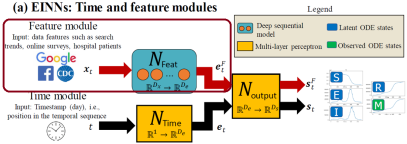

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- 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}): $\frac{1}{N+1} \sum_{t=t_0}^{t_N} \left[e_t - e_t^F\right]^2$

EINN Model

Target model (Feature module)

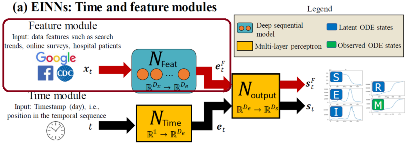

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- 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 module losses

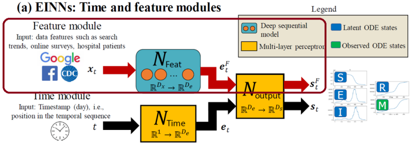

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $e_t \approx e_t^F$ 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

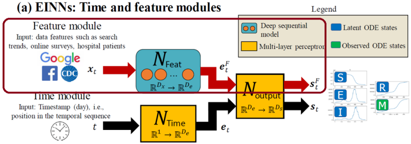

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $e_t \approx e_t^F$ 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

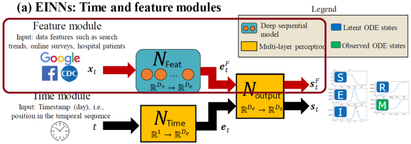

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

HOCHSCHULE MITTWEIDA

EINN Model

Target module losses

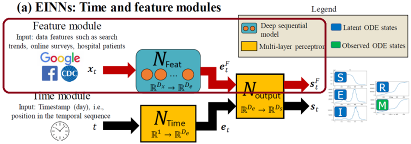

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $\hat{M}_t^F$: 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

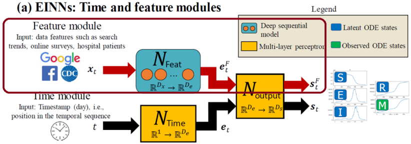

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- $\hat{M}_t^F$: 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

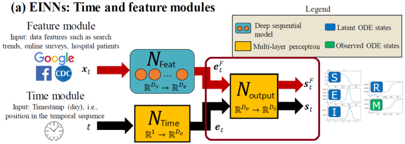

Figure: Cropped model summary showing the training summary. Cropped from: [1]

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

EINN Model

Target module losses

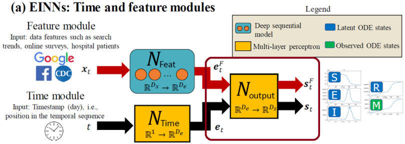

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- N_{output} feed-forward Neural Network
- 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

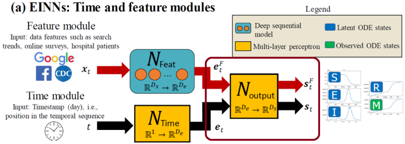

Figure: Cropped model summary showing the training summary. Cropped from: [1]

- N_{output} feed-forward Neural Network
- 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

- Generation: The NN learns directly from ODE generated data [1, 6, 7]
- Regularization: The NN predicts states and parameters of an ODE and learns from the parameters [1, 8, 9]
- Ensembling: Combines predictions of RNN and SEIRM/SIR outputs in one NN for a final prediction [1, 10, 11]
- EINNs-NoGradMatching: skipping $e_t \approx e_t^F$ part [1]

Results

Model	Short-term (1-4 wks) NR1	NR2	ND	Long-term (5-8 wks) NR1	NR2	ND	Trend correlation PC
Task 1: COVID-19 Forecasting (US National + 47 states)							
RNN (GRU+Atten)	1.09	0.50	0.86	1.19	0.53	0.96	0.08
Mechanistic model (SEIRM)	2.35	1.13	1.36	7.14	2.99	3.11	**0.53**
GENERATION	0.79	0.35	0.60	**0.93**	**0.40**	0.74	-0.01
REGULARIZATION	1.05	0.48	0.81	1.19	0.53	0.97	0.09
ENSEMBLING	0.91	0.41	0.68	**0.93**	**0.40**	**0.69**	-0.01
EINNs (ours)	**0.54**	**0.24**	**0.38**	**0.85**	**0.37**	**0.66**	**0.46**
PINN (time module standalone)	0.84	0.38	0.64	**0.93**	**0.40**	0.72	0.24
EINNs-NoGradMatching	**0.64**	**0.29**	**0.49**	0.98	0.43	0.79	0.03
Task 2: Influenza Forecasting (10 HHS regions)							
RNN (GRU+Atten)	0.72	0.38	0.67	1.19	0.51	1.14	-0.03
Mechanistic model (SIRS)	0.72	0.38	0.51	1.16	0.55	0.81	**0.71**
GENERATION	0.76	0.4	0.71	1.21	0.52	1.15	-0.14
REGULARIZATION	1.19	0.64	1.00	1.22	0.54	0.9	-0.45
ENSEMBLING	0.89	0.47	0.77	**0.83**	**0.35**	**0.73**	-0.69
EINNs (ours)	**0.53**	**0.27**	**0.37**	**1.01**	**0.42**	**0.73**	**0.68**
PINN (time module standalone)	0.55	0.29	0.44	1.13	0.48	1.02	-0.47
EINNs-NoGradMatching	**0.53**	**0.27**	**0.38**	1.02	**0.42**	0.76	0.50

Figure: Short-term, Long-term, and Trend results. NR: Normalized RMSE (Lower is better), ND: Normal Deviation (Lower is better) PC: Pearson Correlation (Higher is better). Image taken from: [1]

Results

- Better short and long term prediction than RNN and Mechanistic model (SEIRM)
- Better trend correlation than RNN and sligtly less trend correlation than Mechanistic model (SEIRM)

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.

[6] L. Wang and et al., *Defsi: Deep learning based epidemic forecasting with synthetic information*, 2019.

[7] Sanchez-Gonzalez and et al., *Learning to simulate complex physics with graph networks*, 2020.

[8] J. Gao and et al., *Stan: Spatio-temporal attention network for pandemic prediction using real-world evidence*, 2021.

Bibliography III

[9] N. Gaw and et al., *Integration of machine learning and mechanistic models accurately predicts variation in cell density of glioblastoma using multiparametric mri*, 2019.

[10] A. Adiga and et al., *All models are useful: Bayesian ensembling for robust high resolution covid-19 forecasting*, 2021.

[11] T. K. Yamana and et al., *Individual versus superensemble forecasts of seasonal influenza outbreaks in the united states*, 2017.

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

hs-mittweida.de