Neural-MPC for Control of Type 1 Diabetes

Coletta Emanuele
Gianluca Sperduto

DEPARTMENT OF COMPUTER, CONTROL, AND MANAGEMENT ENGINEERING ANTONIO RUBERTI



SAPIENZA
UNIVERSITÀ DI ROMA

Course:
Control of Comunication and Energy Networks

A.A. 2025/2026

Prof:
Menegatti Danilo
Baldisseri Federico

Table of contents

- Introduction
  - White,Black,Gray-box models
  - Neural-MPC
  - MPC for Diabetes
- Development
  - Ruan and UVA/Padova models
  - Learning a correction
  - Recursive predictions for long horizons
  - Learning the UVA/Padova dynamics
- Conclusions
  - Future work

INTRODUCTION

White,Black,Gray-Box models

Introduction

- White-box models refer to "models" in the usual sense: mathematical representations of a phenomenon, usually derived from first principles
- Black-box models (predictors) are the opposite: representations of the phenomenon entirely derived from data captured from the phenomenon itself
- Gray-box models are a compromise between the two: they are a combination of mechanistic and data-driven models.
    - Uses in climate control of buildings, chemical processes, power systems, robotics

White,Black,Gray-Box models

Black-box models

- Data-driven methods (e.g. Neural Networks) can be used to learn the dynamics of a process from collected data
- And the obtained predictor can be used in a MPC formulation

- Problems: explainability, accumulation of error with multiple consecutive predictions

- Possible solution: inform the predictor about some (possibly, approximated) knownledge of the underlying dynamics (e.g. PINNs)

White,Black,Gray-Box models

Grey-box models

- Also called hybrid, Hammerstein or Weiner models
- They combine an approximated mechanistic model with the generalization capabilities of a data-driven model
  - The data-driven part is used to account for unmodeled nonlinearities, parameter variation

- Different structures to mix mechanistic and data-driven predictions
  - A parallel structure is used in this project

- Data driven approaches: Least Square optimization, Gaussian Processes, CHoKI decomposition, DMD, SYNDy, Neural Networks (RNN, LSTM)

White,Black,Gray-Box models

Grey-box methods

- Some examples of uses:
  - In HVAC systems/climate control of buildings (online estimation of building parameters)
    - Although machine learning methods are rarely used
  - Robotics, especially in conjuction with Reinforcement Learning to learn Residual policies (correction to the control action)
  - Industrial and chemical processes
- Uses of this methodology seem rare in the literature
  - Is it easier to learn a residual dynamics or the whole dynamics?

MPC

- The **Model Predictive Controller (MPC)** is an advanced control strategy that aims to find the best control input using the model to predict the future evolution of the system.
- Require a good model
- Can manage constraints, optimize w.r.t. arbitrary criteria and offer an intrinsic robustness.

$$\min_u \int_0^T (x^T R x + u^T Q u) dt + x^T F x$$

s.t.:

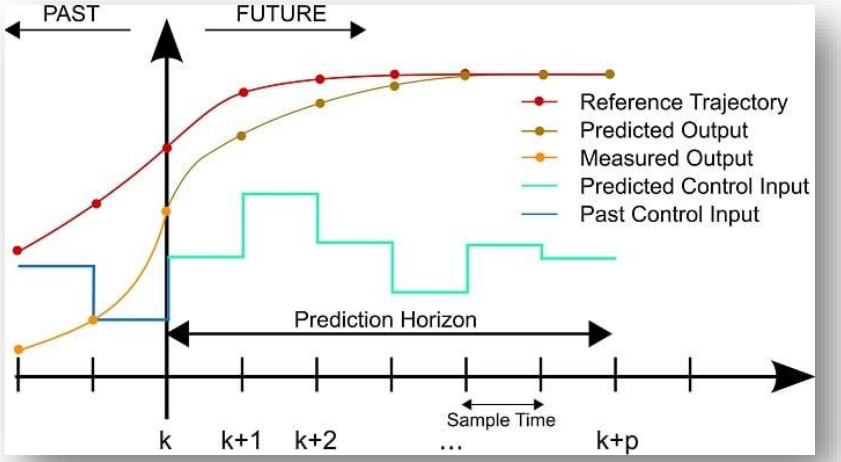
$$\dot{x} = Ax + Bu$$

$$f(x) \leq 0$$

$$h(x) = 0$$

$$g(u) \leq 0$$

$$c(x(T)) = 0$$



MPC for Diabetes

Intro

- **T1DM** is an **autoimmune disease** characterized by the destruction of **pancreatic β− cells**, which are responsible for producing **insulin**
- Insulin is needed for **Blood Glucose** to enter cells and be used for energy
- High risks associated to **hyperglycemia** (**BG ≥ 180 mg/dL**) and **hypoglycemia** (**BG ≤ 70 mg/dL**).
- In the last 50 years, high efforts to automate insulin delivery (**Artificial Pancreas**)
  - **Insulin pumps** to deliver insulin in subcutaneous tissue
  - **Continuous Glucose Monitors (CGM)** to measure glucose levels
  - Closed loop systems (initially PID, now MPC) to automatically compute the insulin dose to be delivered and ensure safety of patients

- Availability of an in-silico simulator, approved by the FDA: **UVA/Padova** simulator
  - The model offered by the simulator is the most complete (and complex)
  - Other models exist (Ruan, Bergman, Hovorka...)

MPC for Diabetes

Challenges faced

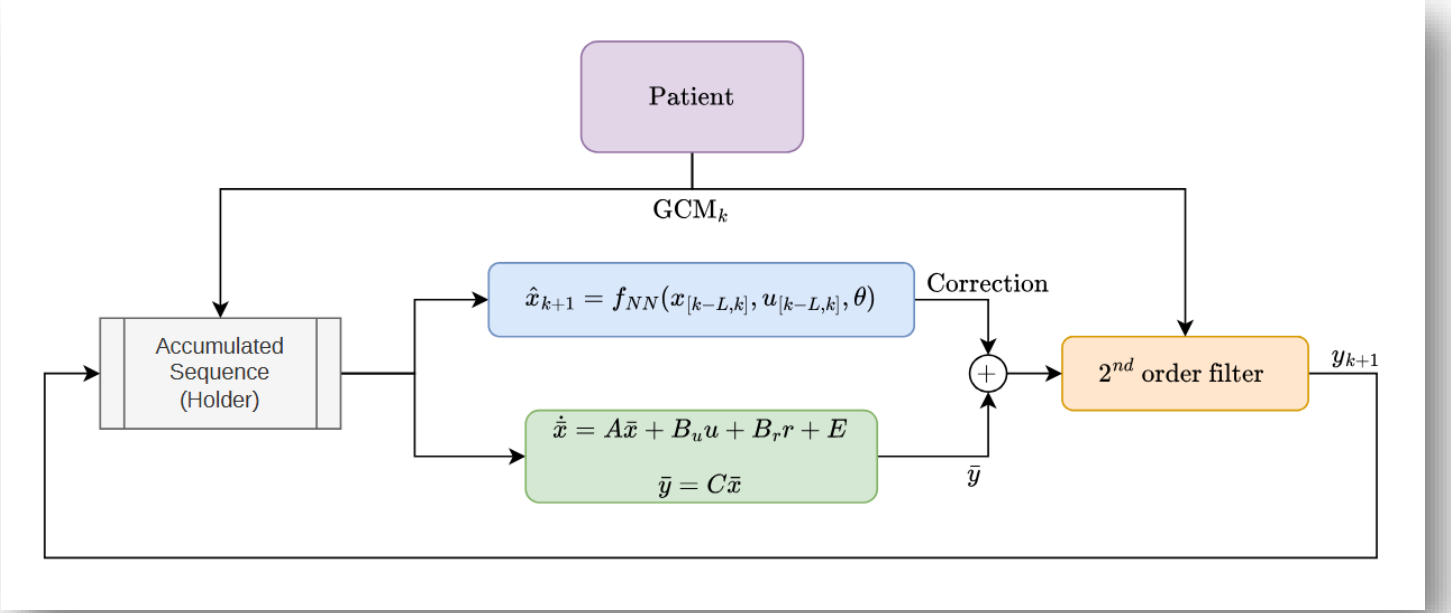
- In most systems **meals** need to be announced
- As usual in MPC, complex models offer better prediction but require greater computational efforts
    - APs are embedded devices with **limited computational capability**

- Handling of variations such as insulin sensitivity **intra-patient and inter-patient**
- Safety constrains to avoid **hypo- and hyper-glyceamia**

MPC for Diabetes

This project

- We develop a **gray-box** prediction model to be used in MPC for control of T1DM
  - We start from the **linear Ruan** model
  - The Ruan model is **augmented with a NN (LSTM)** that predicts unmodeled nonlinearities
  - The network is trained on data generated with the **UVA/Padova** simulator



DEVELOPMENT

Neural Network for predictions

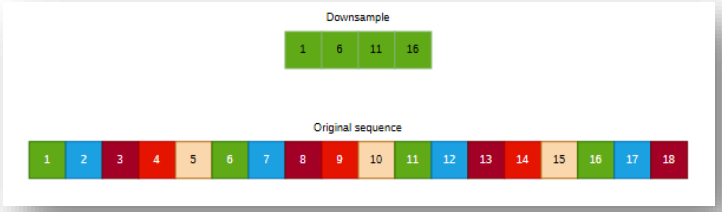
Structure

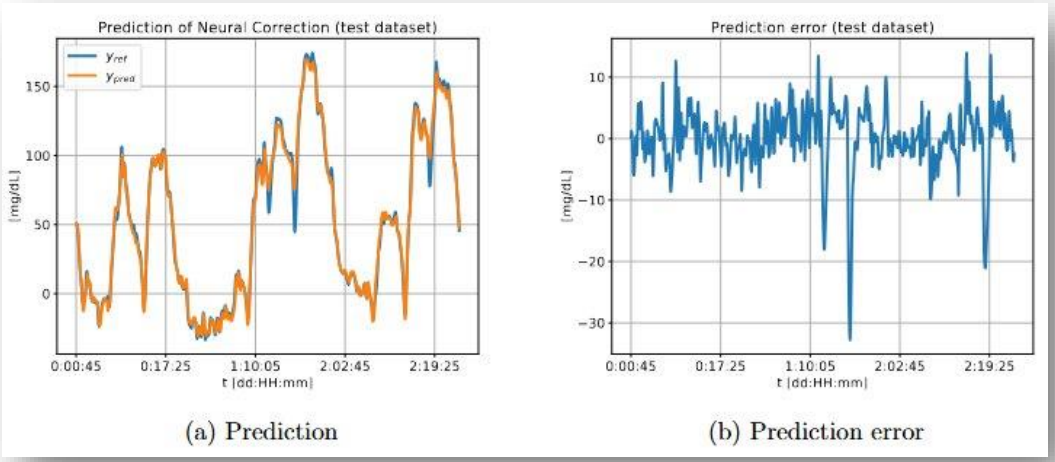
- The Long Short Term Memory (LSTM) is a particular Neural Network with recurrent edges
- Designed for capture long term correlation, learn dynamics and internal state
- Our Design: 2 LSTM layers (300 neurons),
  2 Fully connected layers (300 neurons, ReLU)
- Input: Sequence of 70 samples (5 minutes sampling time - 6h in the past) of insulin, meals (carbohydrate intake) and past measures of CGM coming from UVA/Padova (i.e. patients)
- Output: difference between CGM computed by the Ruan model and UVA/Padova under the same insulin-meal sequence, one timestep in the future
- Training uses early stopping to avoid overfitting, patience of 20 epochs
- Column-wise normalization of data

Down-Sampling

First Approach

- The dataset is composed by CGM measures over 28 days sampled at 1 minutes.
- Down-Sample: Keep samples multiple of 5 minutes and discard the others
  - This is done to match MPC frequency
- 21 days of training, 4 validation, 3 testing





(a) Prediction (b) Prediction error

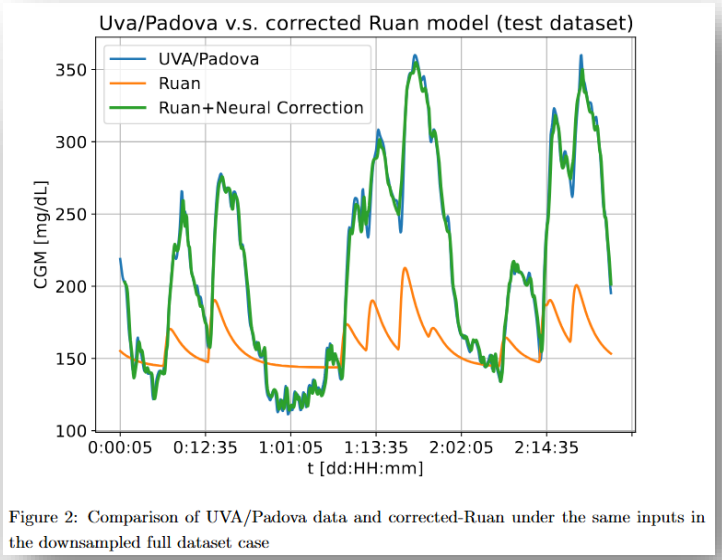
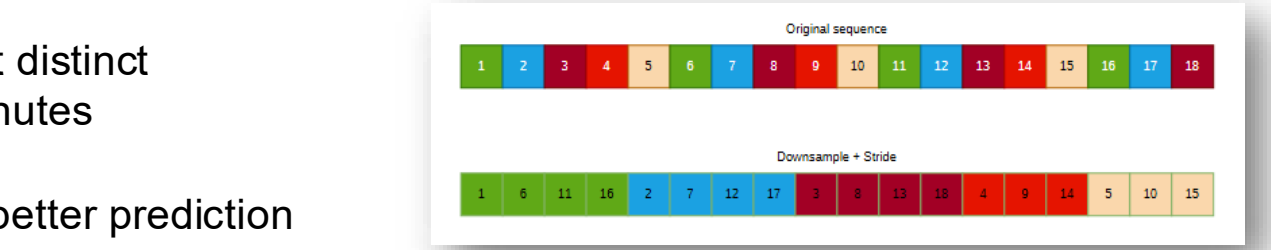


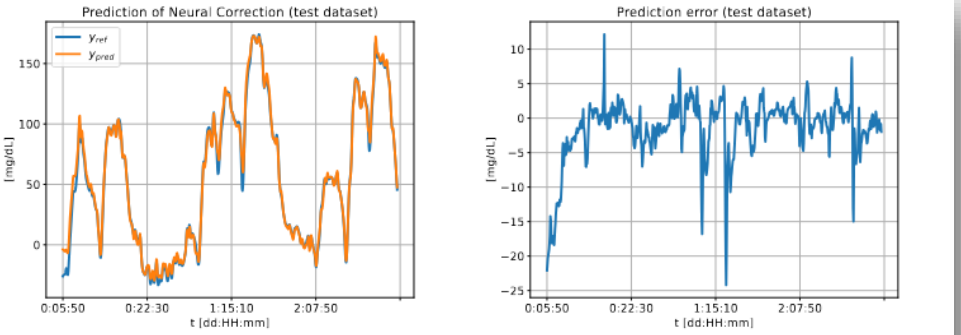
Figure 2: Comparison of UVA/Padova data and corrected-Ruan under the same inputs in the downsampled full dataset case

Using Stride

Downsampling + Stride

- From the same data, extract distinct sequences sampled at 5 minutes
- More training data leads to better prediction capabilities (lower error)





(a) Prediction (b) Prediction error

Figure 4: Prediction and prediction error on test dataset in the downsampled with strides case

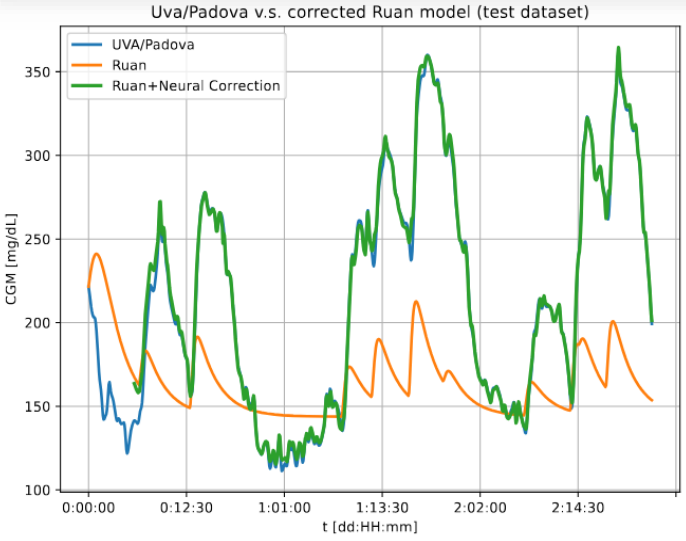
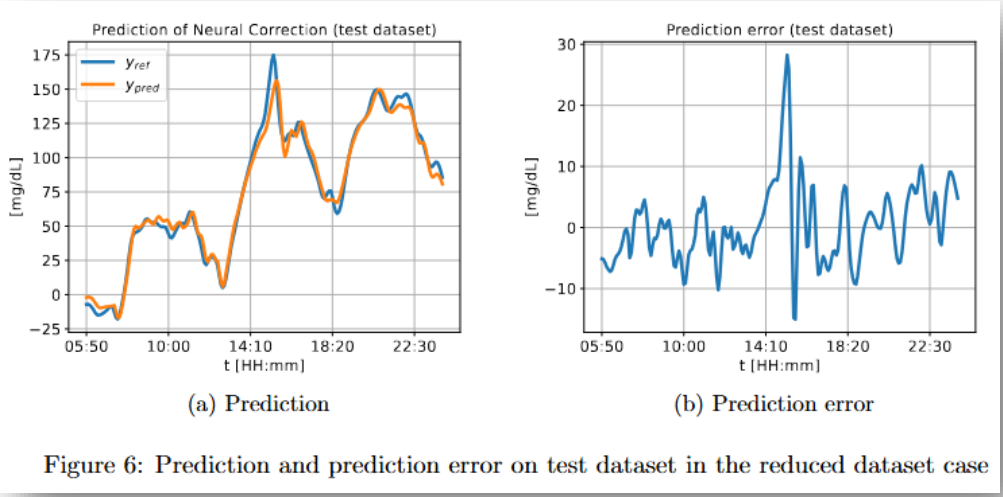


Figure 5: Comparison of UVA/Padova data and corrected-Ruan under the same inputs in the downsampled with strides case

Reduced Dataset

Can the network be trained with less data?

- Ideally, the patient would be hospitalized for one week while recording CGM, carbohydrate intake and given appropriate insulin (manually or with other AP)
- 5 days used for training, 1 for validation, 1 for testing



(a) Prediction (b) Prediction error

Figure 6: Prediction and prediction error on test dataset in the reduced dataset case

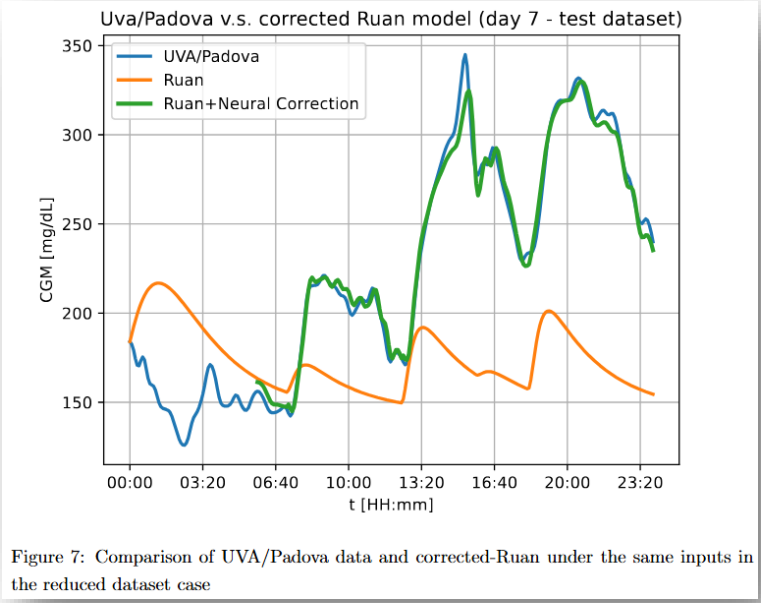
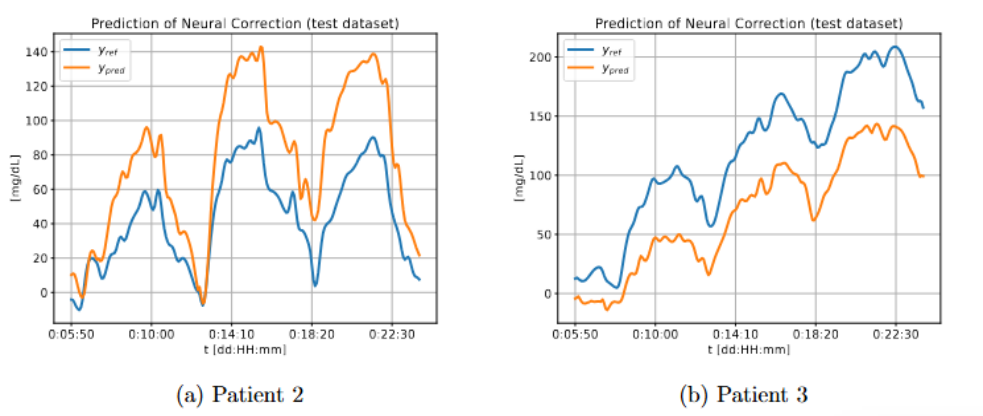


Figure 7: Comparison of UVA/Padova data and corrected-Ruan under the same inputs in the reduced dataset case

Generalization to different patients

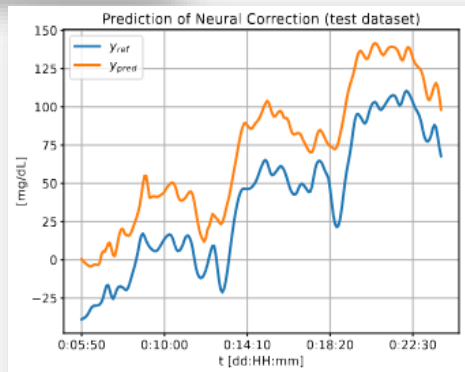
When applied on different patients an error appears (~ constant/ramp)

- The network is trained on data from one patient, Ruan model parameters estimated from the same data

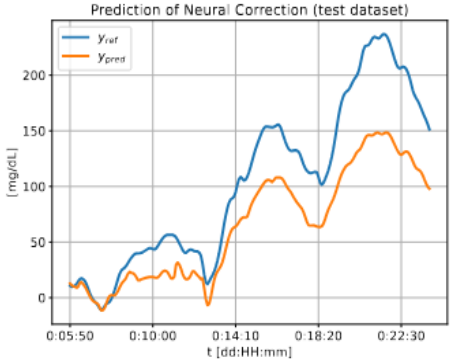


(a) Patient 2

(b) Patient 3



(c) Patient 4



(d) Patient 5

Bias as disturbance

Second order filter to estimate prediction error

- Seeing only UVA/Padova data, the network cannot **learn the bias** between it and the Ruan model

- We consider this **bias as a ramp output disturbance (at worst)** and use a 2nd order filter to estimate it and correct for it with a feedforward action

$$\dot{r} = \begin{bmatrix} \dot{r}_1 \\ \dot{r}_2 \end{bmatrix} = \begin{bmatrix} r_2 \\ K_P(e - r_1) + K_D(\hat{\dot{e}} - r_2) \end{bmatrix}$$

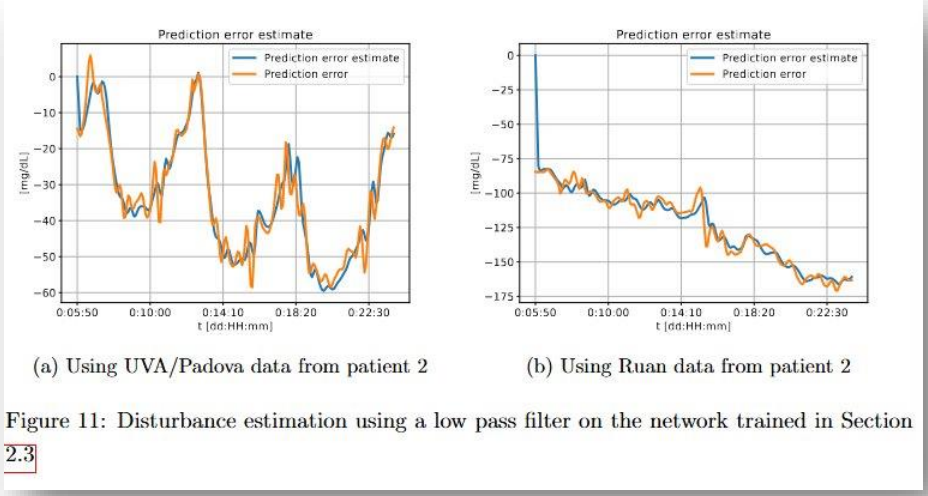
$$r(k+1) = \begin{bmatrix} r_1(k+1) \\ r_2(k+1) \end{bmatrix} = \begin{bmatrix} r_1(k) + \Delta T \cdot r_2(k) \\ r_2(k) + \Delta T \cdot \left[K_P\Big(e(k) - r_1(k)\Big) + K_D\Big(\hat{\dot{e}}(k) - r_2(k)\Big) \right] \end{bmatrix}$$

Where the derivative is approximated as the finite difference $\hat{\dot{e}}(k) = \frac{e(k) - e(k-1)}{\Delta t}$.

Bias as disturbance

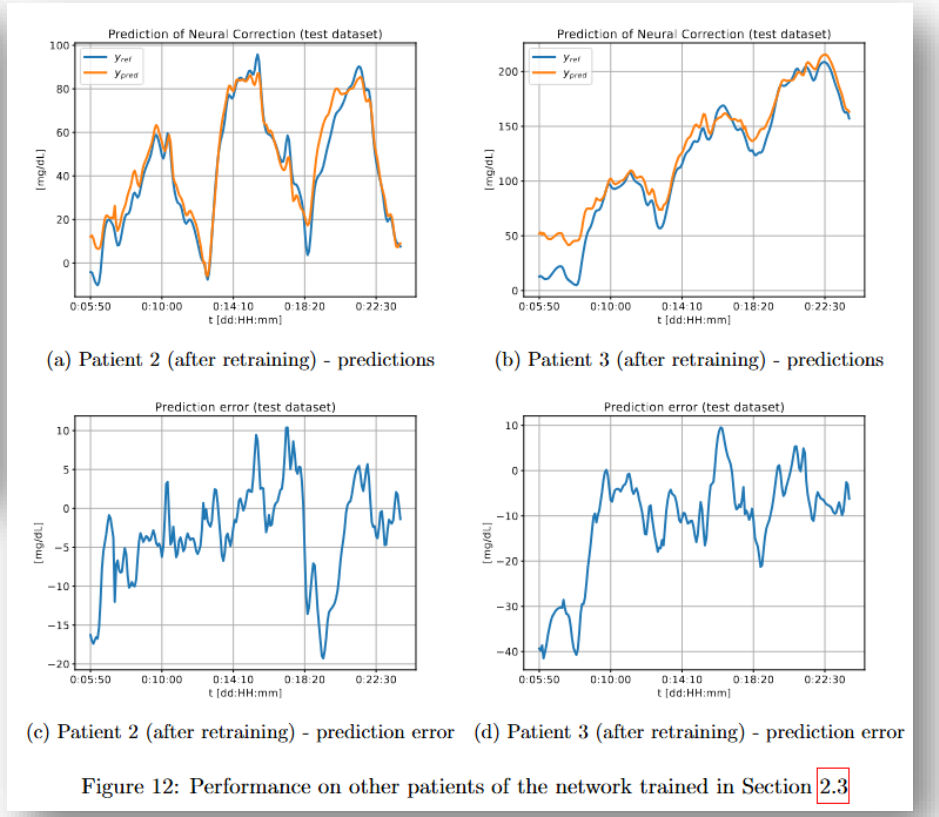
Second order filter to estimate prediction error

- Even without optimal tuning, the filter can reconstruct the disturbance very well



(a) Using UVA/Padova data from patient 2 (b) Using Ruan data from patient 2

Figure 11: Disturbance estimation using a low pass filter on the network trained in Section 2.3



(a) Patient 2 (after retraining) - predictions (b) Patient 3 (after retraining) - predictions

(c) Patient 2 (after retraining) - prediction error (d) Patient 3 (after retraining) - prediction error

Figure 12: Performance on other patients of the network trained in Section 2.3

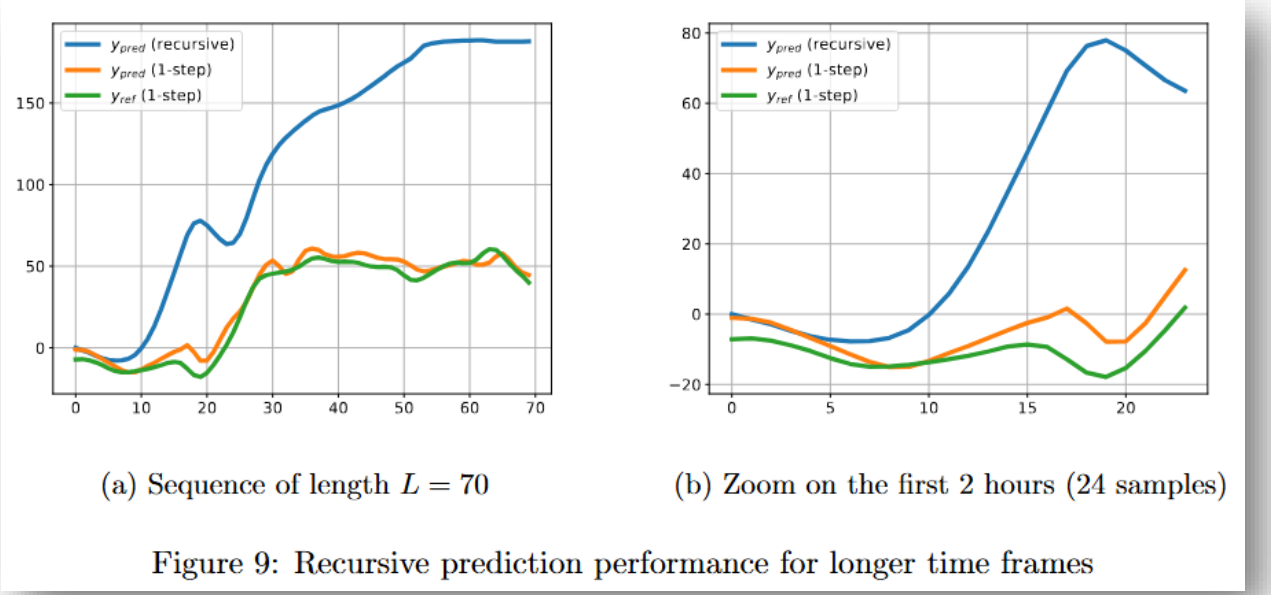
Refine training using other patients data

Chained Predictions

The network only predicts 1-step-ahead

- **Chain multiple predictions** to account for longer horizons, to be used in MPC
- Ruan is simulated using the **current measurement as initial conditions**
- The correction is added inside the cost function, under the currently computed inputs

- **Bad results:** error accumulates and the prediction diverges quickly



(a) Sequence of length $L = 70$

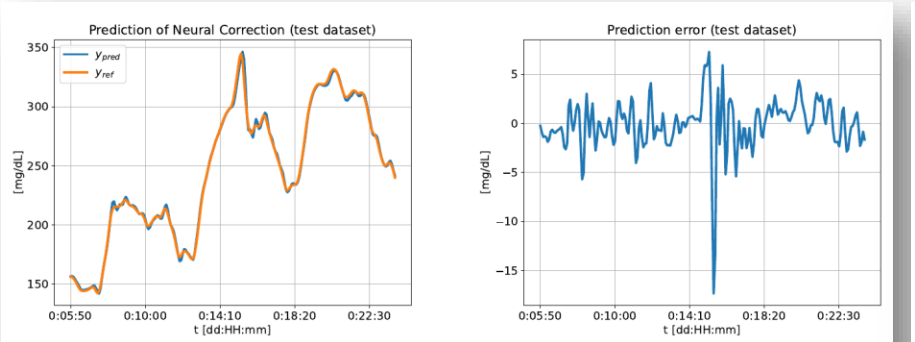
(b) Zoom on the first 2 hours (24 samples)

Figure 9: Recursive prediction performance for longer time frames

Learning UVA/Padova

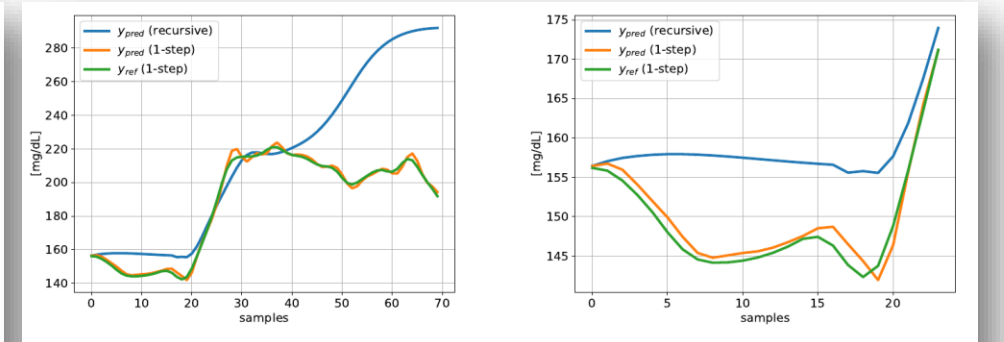
The network must learn the Ruan dynamics, but only has the output

- Would the network perform better if it only **predicted UVA/Padova** 1-step-ahead?
- **Almost perfectly learns** the UVA/Padova dynamics, even with reduced dataset
  - Results consistent with existing literature
- Chained predictions **diverge slower, but have larger initial error**



(a) Prediction (b) Prediction error

Figure 13: Prediction and prediction error when learning the UVA/Padova model



(a) Sequence of length $L = 70$ (b) Zoom on the first 2 hours (24 samples)

Figure 14: Recursive prediction performance for longer time frames

MPC Formulation

Proposal of MPC formulation with NN+filter

- Using the correction model, filter and saturations.

$$
\begin{aligned}
&\min_{u} \quad J(y_{[k,k+h]}, u_{[k,k+h]}, t) \\
&\quad s.t.: \\
&\bar{x}_{k+1} = Ax_k + B_u u_k + B_r r_k + E \\
&\hat{x}_{k+1} = f_{NN}(x_{[k,k-L]}, u_{[k,k-L]}, \theta) \\
&y_{k+1} = F_2(C\bar{x}_{k+1} + \hat{x}_{k+1}, y_{\text{measured}}) \\
&\\
&m_x \leq x_k \leq M_x \\
&m_u \leq u_k \leq M_u
\end{aligned}
$$

$$
J(y_{[k,k+h]}, u_{[k,k+h]}, t) = \sum_{i=k}^{k+h} Q(y^{\star}(t) - y_i)^2 + R(\bar{u}_i - u_i)^2
$$

Comparison

Training and testing performance comparison across all methods (1 patient)

Training	Training loss	Validation loss	Error mean	Error STD
21-day training (downsample)	18.805	24.010	−3.383	5.918
21-day training (downsample+stride)	16.529	3.550	−6.513	4.402
5-day training (downsample+stride)	13.500	36.715	−2.232	7.391
5-day training (downsample+stride; UVA/Padova)	10.931	4.433	−0.108	2.573

CONCLUSIONS

Conclusion

Recap

- In this project a **neural network (LSTM)** was trained to predict the mismatch between a linear (**Ruan**) and full (**UVA/Padova**) model for insulin dynamics
- The network has **good prediction** capabilities, even when trained on small datasets (**5 days of training**)
- The network **can also generalize** to patients unseen during training with some precaution
- Performance degrade quickly when chaining multiple 1-step predictions to account for longer MPC horizons
- The same NN can also learn the entire UVA/Padova dynamics, with similar results

Conclusion

Future work

- Implementation of **MPC** with the UVA/Padova simulator to validate the methodologies used:
  - Is the prediction error low enough for closed loop control?
  - Does the error in chained predictions degrade performance in MPC, or can it be worked around (e.g. with **weights in the cost function**, or by correcting each step of the predictionor **reducing horyzon**)?
  - Which one of the approaches offers the best performance/quantity of data compromise in closed loop?
  - Can such closed loop system network cope with **unannounced or wrongly announced meals**?

Thank you for your attention!