DIPLOMARBEIT

Enhancements in CGM Forecasting: Robustness Against Domain Shifts and Safer Predictions

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Technische Mathematik

eingereicht von

András Sass
Matrikelnummer 01527290

ausgeführt am

Chair of Information Management
ETH Zürich

unter der Betreuung von

Prof. Dr. Michael Feischl

unter der Beratung von

Ph.D. Simon Föll

Wien, am 17.10.2023

THESIS

Enhancements in CGM Forecasting: Robustness Against Domain Shifts and Safer Predictions

written at

Chair of Information Management
ETH Zürich

under the supervision of

Prof. Dr. Michael Feischl

under the advisement of

Ph.D. Simon Föll

by

András Sass

[redacted]

Vienna, on 17.10.2023

Kurzfassung

Die weltweite Zahl der Diabetiker wächst, mit geschätzten 536 Millionen betroffenen Personen im Jahr 2021 und Prognosen von bis zu 783 Millionen bis 2045. Eine genaue Glukoseprognose ist im Diabetes-Management von großer Bedeutung, da diese rechtzeitig vor gefährlichen Glukoseabweichungen warnt und somit Komplikationen verhindert. Obwohl zahlreiche Glukosevorhersagemethoden existieren, sind diese selten auf klinisch relevante Vorhersagen optimiert und die Herausforderungen im Zusammenhang mit Verteilungsverschiebungen über Patientenpopulationen aufgrund von Faktoren wie neuen Diabetesbehandlungen führen zu einer mangelnden Robustheit. Diese Arbeit adressiert diese kritischen Lücken und schlägt verbesserte Modelle vor, welche klinisch bedeutsame Fehler minimieren als auch robuster gegenüber sich verändernden Verteilungen sind.

Unser diverser Datensatz beinhaltet unterschiedliche Diabetes-Profile. Wir nutzen long short-term memory (LSTM) und attention-basierte Ansätze für probabilistische continuous glucose monitoring (CGM)-Prognosen und implementieren den Parkes error grid (PEG)-loss zur Reduktion klinisch relevanter Fehler. Durch die Integration von gated-domain-units (GDUs) wird die Modellrobustheit erhöht. Die Bewertung erfolgt sowohl über klassische Metriken wie negative log-likelihood (NLL) und root-mean-square error (RMSE) als auch über den PEG.

Die Verwendung des PEG-loss resultiert in einer signifikanten Reduktion von medizinisch relevanten Vorhersagefehlern bei beiden Architekturen. Der Einsatz von GDUs verbessert die Performance der Vorhersagemodelle für Individuen mit zuvor nicht gesehenen Diabetes-Behandlungen. Die GDU-Modelle zeigen eine verbesserte Robustheit gegenüber Verteilungsverschiebungen der Daten und übertreffen Ensemble-Modell-Benchmarks. Diese Modelle erhöhten auch die Interpretierbarkeit der Daten und beleuchten verschiedene Subdomänen innerhalb des Feature-Raums.

Insgesamt stellt diese Arbeit einen bedeutenden Schritt zur Entwicklung von CGM-Prognosemodellen dar, die nicht nur eine hohe technische Leistung aufweisen, sondern auch klinischer Signifikanz gerecht werden. Obwohl die Modelle lobenswerte Domänengeneralisierungsfähigkeiten aufweisen und sie an vielfältige Diabetes-Behandlungen anpassbar machen, sind sie nicht ohne Limitationen. Die inhärenten Komplexitäten der CGM-Daten stellen einige Herausforderungen dar, welche zusätzliche Aufmersamkeit benötigen. Dennoch ebnen die Ergebnisse den Weg für vielversprechende zukünftige Forschungen und praktische Anwendungen und könnten das Diabetesmanagement für eine vielfältige Bevölkerung verbessern.

Abstract

The global diabetic population is increasing, with an estimated 536 million individuals affected in 2021 and projections rising to 783 million by 2045. Accurate glucose forecasting is paramount in diabetes management, as it provides timely warnings against dangerous glucose deviations, thus preventing complications ranging from retinopathy to death. While numerous glucose prediction methods exist, they are seldom optimized towards clinically relevant predictions and the challenges related to distribution shifts across patient populations due to factors like new treatments lead to a lack of robustness. This research addresses these critical gaps, proposing enhanced models that are both safer, by minimizing clinically consequential errors, and more robust, accommodating shifting distributions in glucose monitoring data.

In our study, we use a diverse dataset of individuals with varying diabetes profiles. We employ an LSTM and an attention-based architecture for probabilistic CGM forecasting. We introduce the PEG loss to train deep learning architectures towards making less clinically relevant errors. Integration of GDUs ensures model robustness. Model performance is assessed using both classical metrics, i.e., NLL and RMSE, as well as clinical error evaluations based on the PEG.

Our novel PEG loss function, tailored for reducing clinically significant errors, demonstrates its efficacy when applied to both LSTM and attention-based models, achieving a marked reduction in clinically relevant prediction errors. This is achieved without a significant increase in computational complexity. The adaptability offered by the PEG loss enables performance optimization across various clinical scenarios. Furthermore, using gated-domain-units (GDUs) enhances domain generalization capabilities to new treatments. The integrated gated-domain-units (GDU) models exhibit improved robustness against data distribution shifts, surpassing ensemble model baselines in performance. These models also enrich model interpretability, shedding light on diverse subdomains within the feature space.

Overall, the research presents a significant step towards developing CGM forecasting models that not only offer high technical performance but also adhere to clinical significance. While our models show commendable domain generalization capabilities, making them adaptable to diverse diabetes treatments, they are not without limitations, particularly concerning the inherent complexities of CGM data and challenges in model design as well as evaluation. Nonetheless, the findings pave the way for promising future research and practical applications, potentially improving diabetes management for a diverse population.

Acknowledgement

Words cannot express my gratitude to my supervisor, Ph.D. Simon Föll, for his invaluable patience and feedback. I want to thank Ph.D. Eva van Weenen, who generously provided knowledge and expertise. I also could not have undertaken this journey without my supervisor from TU Wien, Prof. Dr. Michael Feischl, who generously guided my efforts from my home university.

I am also grateful to my parents, girlfriend, and my brother. Their belief in me has kept my spirits and motivation high during this process.

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 17.10.2023

András Sass

Contents

1. Introduction

1.1. Motivation

Diabetes mellitus, a chronic condition characterized by an inability of the body to sufficiently produce or effectively utilize insulin, is an increasingly prevalent health concern [1]. In 2021, the International Diabetes Federation reported an estimated 536 million individuals aged between 20 and 79 years old affected by this condition, with projections rising to 783 million by 2045 [2]. Effective management of diabetes necessitates consistent maintenance of blood glucose (BG) concentration within specified limits through behaviors such as BG monitoring, insulin administration, and dietary control. Deviations from these thresholds can lead to hypo- or hyperglycemia, conditions which predispose individuals to a spectrum of complications, including retinopathy, nephropathy, neuropathy, coronary heart disease, cerebrovascular disease, peripheral vascular diseases, unconsciousness, and death [3].

Recent advancements in therapeutic strategies seek to enhance glycemic control, with notable efforts directed towards the development of an artificial pancreas (AP)—a fully autonomous glucose management system designed to maintain BG and consequently mitigate associated health risks [4, 5]. CGM sensors, which enable the continuous tracking of interstitial fluid glucose concentration, are integral components of these modern diabetes treatments. Despite their utility, most treatments relying on CGM data suffer from latency, inherent in the delayed response of glucose and insulin action, and the physiological lag of the CGM readings behind actual BG levels [4]. Thus, the imperative need for accurate CGM forecasting models emerges, aimed at reducing treatment response time and maintaining BG within safe ranges as elucidated by Ma et al. [5].

The first problem we are tackling in this work is the minimization of clinically consequential CGM prediction errors, such as falsely predicting the rise of blood glucose when it is already dangerously low. Avoiding clinically consequential errors results in overall safer CGM predictions. While considerable work has been invested in the construction of glucose prediction models leveraging CGM data, the primary focus has largely been on enhancing technical performance rather than mitigating clinically consequential errors [6, 7]. Erroneous forecasts in conditions of hypo- and hyperglycemia can result in grave health outcomes [3], which underscores the importance of our endeavour to develop a framework capable of identifying models that minimize such critical errors.

Secondly, we are addressing the problem of distribution shifts in the population's CGM data, thus making prediction models more robust and more widely applicable. As stated by Vapnik [8], statistical learning algorithms fundamentally depend on the assumption that source and target data are independent and identically distributed (i.i.d.). Thus, an agent trained on

source data typically experiences substantial performance declines when confronted with target data that is not identically distributed [9]. This scenario is called distributional shift or domain shift. A recent example of a case in medicine where a domain shift resulted in a significant performance drop is described by Wong et al. [10]. A proprietary sepsis prediction model suffered substantial performance losses when applied in hospitals that were not part of the development process. Other cases of domain shifts in medical machine learning applications lead to the publication of potential sources of distribution shifts in healthcare settings [11]. One of the potential sources is the introduction of new treatments or standards of care. Hence, in the context of the ongoing evolution of therapeutic measures for individuals with diabetes, it is reasonable to assume that the population's CGM trajectories exhibit a shifting distribution. Existing models, primarily trained and tested using CGM data central to the used datasets, are vulnerable to distribution shifts induced by novel treatments. This necessitates rigorous evaluation and enhancement of model robustness against distribution shifts when developing glucose prediction models to prevent potential performance drops that could have serious medical implications for large diabetic populations. Our work, thus, prioritizes the development of models with robust generalizability to unseen domains, such as novel treatment protocols, underscoring the critical importance of our research in this evolving landscape of diabetes management.

1.2. Related work

Extensive research has been conducted in recent years focusing on glucose prediction, employing a range of techniques and methods [6, 7, 12–24]. Existing literature categorizes CGM forecasting models into three predominant categories: physiology-based (knowledge-based), data-driven (empirical-based), and hybrid approaches—a combination of the previous two [6, 7]. Recent findings suggest that data-driven models consistently outperform white-box models with well-structured physiological parameters, even when individualized, indicating their superiority in glucose prediction tasks [24].

The dynamics of BG are influenced by several factors including carbohydrate intake, insulin administration, physical activity, and stress [5]. The integration of such factors as additional inputs improves the performance of prediction models [25]. However, the acquisition of such information imposes an additional burden on users in real-world scenarios [23], motivating a focus on prediction models that rely solely on CGM data [7]. Among these, the most commonly employed for CGM forecasting are neural network models. LSTM models, a particular type of recurrent neural network (RNN) specifically designed for sequence modeling, have been widely utilized, outperforming traditional machine learning models like auto-regressors, random forests, and support vector machines [18–20, 23]. Attention-based architectures, which have demonstrated superior performance in various sequence-to-sequence modeling tasks [26], have led to the design of specific architectures for time series forecasting [27–29]. These architectures, however, have yet to be employed in CGM forecasting.

Error-grid-analysis, developed to assess the accuracy of BG meters such as CGM devices, is commonly used for patient self-measurement [30, 31]. In this method, self-monitoring device measurements are compared to the actual BG levels obtained via blood samples, with

the resulting two-dimensional points falling within specific risk zones within a grid. This analysis also facilitates the assessment of the clinical relevance of CGM forecasting models [7], thus providing a methodology to evaluate model errors in terms of clinical severity ex post. To the best of our knowledge, only one previous study [15] has attempted to measure and improve a model's clinical performance during model training, aiming to identify models with increased clinical relevance. However, the Clarke error grid (CEG) utilized in their methodology is outdated, suggesting the need for newer error grids such as the PEG [31] for a more accurate model identification process.

In recent years, substantial research has been conducted on domain generalization (DG) in statistical learning, leading to the development of numerous methods to increase machine learning models' out-of-distribution (OOD) test performance, and thereby enhancing their robustness against domain shifts [9]. Ensemble learning is a commonly studied DG method, wherein a prediction is derived from an aggregation of multiple learners' predictions. This method has demonstrated improved OOD performance in several tasks [32, 33]. Notably, the GDUs developed by Föll et al. [33] offer a novel approach for averaging predictions, based on the premise that distributions are composed of elementary distributions, and that weighted averaging should consider a given sample's similarity to learnable domains representing these elementary distributions [33]. This method outperformed state-of-the-art DG methods across a variety of DG tasks. Within the realm of CGM forecasting, however, the problem of domain shifts has received limited attention. While transfer learning methods proposed in Yu et al. [16] and Luo and Zhao [17] aim to adapt models to new individuals for whom only limited CGM data is available, they are not concerned with the issue of large-scale CGM data domain shifts in the population of diabetics that can occur with the introduction of new treatments like a novel insulin pump or type of insulin. Our analysis of prior works shows a research gap, which is the identification and reduction of performance drops of CGM prediction models due to large-scale domain shifts resulting from the development of diabetes treatments.

1.3. Research questions and approach

Probabilistic forecasts play a vital role in quantifying the uncertainty in predictions and are key components of decision-making [34, 35]. Therefore, predicting future CGM levels involves not only generating accurate forecasts but also understanding the model's certainty about each prediction. We consider probabilistic forecasts [20, 29], which take the form of probability distributions over future quantities, thereby providing insight into the model's certainty about its predictions. Aliberti et al. [18] highlighted that the common practice until now is to calibrate models to individuals. While this individualized approach might lead to performance gains for some individuals, it requires a significant amount of individual data and individual calibration, and also carries the risk of overfitting. Thus, our choice is to train our models on a large heterogeneous dataset that includes individuals of various ages, using different devices, and with different types of diabetes and treatments. According to [12, 18], this approach leads to more robust models that can be applied to new individuals immediately.

This thesis aims to bridge the research gaps identified in CGM forecasting, as substantiated by the following two research questions:

RQ1 How can we integrate the clinical severance of erroneous predictions into a loss function for CGM forecasting deep learning models and to what extent does this loss reduce the amount of clinically severe errors?

RQ2 To what extent can existing robust deep learning methods be utilized to make existing CGM forecasting models robust against distribution changes resulting from the population's diabetes treatment?

To address the first question, we develop a novel loss function, the PEG loss. This loss function aligns with the clinical severity of a model's prediction errors. Our approach is founded on the idea of optimizing the models towards making fewer clinically severe errors, given that neural networks' training procedure is grounded on gradient descent variants. We apply the PEG loss to the state-of-the-art LSTM architecture for CGM forecasting from [20] and an attention-based architecture optimized for time series forecasting [29] to measure the change in the amount of clinically relevant errors. We also measure the added complexity and classical forecasting metrics with and without the novel PEG loss.

The second research question is addressed by enhancing the DG capabilities of our models using GDUs. Specifically, we test our models' generalization ability in situations where no training data is available from individuals undergoing a certain treatment type. We consider several modes of training for the GDU models and compare the generalization performances to a ensemble model baseline. Finally, we measure the added complexity and interpret the learned GDU bases.

The structure of this thesis is as follows: Chapter 2 introduces the dataset, models, metrics, and training procedures. Chapter 3 presents the design of the experiments and their results, along with an ablation study. Chapter 4 offers an interpretation of the results and outlines the limitations. Finally, Chapter 5 concludes the thesis and suggests future research directions.

2. Materials and methods

This chapter presents all materials and methods used in this work. Specifically, we will describe the dataset and preprocessing, the CGM forecasting task, models, loss functions as well as the training and evaluation procedures used.

2.1. Glucose data from multiple domains

2.1.1. Data description

The dataset analyzed in this study comprises a total of 29,371 days of raw CGM readings, from 370 individuals diagnosed with diabetes mellitus. The Department of Diabetes, Endocrinology, Nutritional Medicine, and Metabolism at Inselspital, Bern, Switzerland, collected this data during four open-label field studies from 2013 until 2021. Table A.1 describes and lists the individual participants' characteristics.

For *inclusion* in the studies, participants needed to be older than 16 years and diagnosed with diabetes mellitus. The studies allowed participants to take part in multiple trials, and each participant was equipped with a CGM sensor. Participants' measurements were *excluded* from the final dataset if they failed to provide CGM readings for at least 50% of the data collection period.

The data was organized into four domains, based on treatment type, as determined by Inselspital's medical staff. These treatment types are basal insulin only (BI), multiple daily injections (MDI), continuous subcutaneous insulin infusion (CSII), and artificial pancreas (AP) treatments, detailed in A.1. As evident from Figure 2.1 and Table 2.1, the CGM data exhibits variance across these treatment types. Despite some similarities,

Table 2.1.: CGM data characteristics across domains. The four domains display clinically relevant differences in their characteristics, with the AP domain differing the most from the other domains. The largest values are highlighted in bold. (BI = basal insulin only, MDI = multiple daily injections, CSII = continuous subcutaneous insulin infusion, AP = artificial pancreas)

	all	**BI**	**MDI**	**CSII**	**AP**
days of data	29371	3892	**12873**	9229	3377
average CGM	8.85	**9.06**	8.97	8.81	8.23
glucose variability	3.61	**3.77**	3.76	3.54	2.86
time in range [%]	35.53	62.24	62.31	64.44	**75.32**
hypoglycemia [%]	3.86	3.78	**4.35**	3.92	1.95
hyperglycemia [%]	31.67	**33.98**	33.34	31.64	22.73

structural differences are discernible, with the AP domain diverging most significantly from the others. As patients transition to new, previously unencountered treatment types, the importance of developing forecasting systems robust to these distribution shifts is underscored. Hence, our study emphasizes analyzing forecasting models for their robustness against distribution shifts.

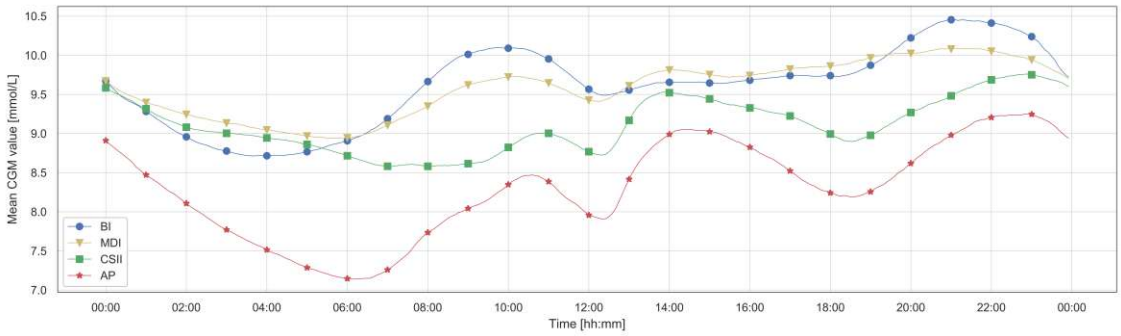


Figure 2.1.: Average daily CGM curves across domains. As new treatments are a potential source of distributional shifts [11], the apparent difference in average daily CGM trajectories supports the claim that distributional shifts are present in the CGM data. We omitted further methodology to indicate differences in distributions, such as the Kolmogorov-Smirnov test or the Wasserstein metric, owing to sufficient visual evidence and the expertise of Inselspital's medical staff. (BI = basal insulin only, MDI = multiple daily injections, CSII = continuous subcutaneous insulin infusion, AP = artificial pancreas)

2.1.2. Data preparation

Due to the data collection under field conditions and the large size of the dataset extensive data preparation was necessary. Through data exploration five types of artifacts were identified: Scans, backward time jumps, pressure-induced sensor attenuations (PISAs), missing measurements, and irregular sampling times. These artifacts were addressed with the following different strategies maximizing data quality whilst minimizing data loss.

Scans. Individuals equipped with a FreeStyle Libre sensor had the capability to execute unscheduled measurements, referred to as *scans*, in addition to the regular, periodic measurements taken by the CGM sensor. The introduction of these scans led to discrepancies within the data sequence. This manifested as CGM measurements that did not align with the surrounding regular readings. At times, these inconsistencies even surpassed physiological feasibility thresholds, indicating readings that were not just irregular but also potentially implausible. Due to these issues, we made the decision to exclude these scans from our analysis. Consequently, this decision led to a data volume reduction of 3.3%.

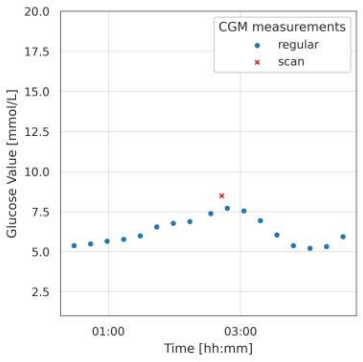


Figure 2.2.: Example of an excluded inconsistent scan measurement.

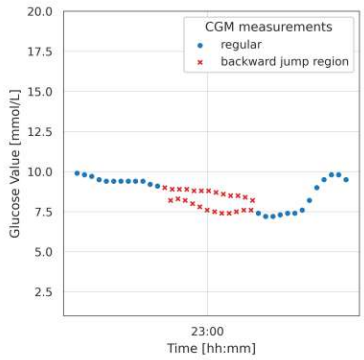


Figure 2.3.: An instance of a backward time jump leading to overlapping time series. Such artifacts constitute 0.7% of the total measurements and were removed during data preparation.

Backward time jumps. Medtronic sensor-derived CGM readings were not chronologically sequenced by their respective timestamps, but by their recording order. This sequence irregularity, attributed to factors such as daylight saving time, travel across time zones, or device time setting errors, resulted in overlapping time series. Backward time jumps, i.e., measurements timestamped earlier than their preceding reading, were identified. Together with their immediately previous readings, these instances established time intervals in which measurements reported CGM levels for different moments. These intervals were removed from the dataset to maintain the time series' sequential integrity, resulting in a 0.7% data volume reduction.

Pressure-induced sensor attenuation (PISA). CGM measurements are prone to artifacts, resulting from pressure applied to the site of the sensor [36]. Generally, the start of a PISA is characterized by a sudden decrease in CGM levels that violate physiological rate-of-change limits. The end of a PISA generally occurs at least 15 minutes later and has a negative rate of change. Several methods to identify PISAs exist [37–39] and all of them come with their respective (dis-)advantages. Considering that a model in production should be able to perform the same data cleaning steps as during model training, the cleaning algorithm has to be able to classify each new CGM reading from a live data stream as being not PISA, the onset of PISA, or the end of PISA, all the while not excluding a prohibitive amount of measurements. The algorithm from Baysal et al. [38] fulfills these criteria and has therefore been chosen in this case. The detection algorithm uses two sets of rules to detect the entering and leaving of a PISA event. From the four modes of parameter settings, which define the cleaning algorithm's aggressiveness, the *nominal* set of parameters was used for a balanced trade-off between the amount of true and false positive PISAs classifications (for details see A.2). By applying this cleaning method 5.1% of measurements were excluded from the dataset.

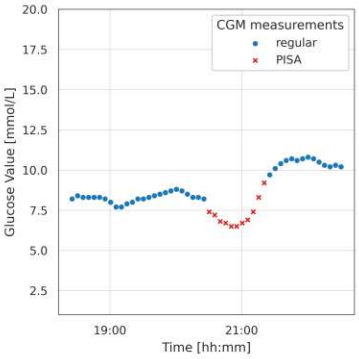


Figure 2.4.: Example of a detected PISA. The detected PISAs were removed, resulting in CGM readings that are representing the true BG of the individuals more reliably and a data loss of 5.1%

Missing measurements. By nature and through the previous cleaning steps, temporal gaps between the CGM measurements were present. To ensure informative inputs and measurable targets for modeling, temporal gaps larger than one hour were marked. The data was then cut into windows with a duration of 24 hours such that the marked temporal gaps were not included in any of the resulting 24-hour windows, ensuring that the windows did not contain temporal gaps larger than one hour. The time of each window's start was randomized to ensure that the samples represent the population's CGM trajectories throughout the whole

day. Each window represents a sample encapsulating both the input and the subsequent target period.

Irregular sampling times The CGM devices produced measurements at irregular sampling times as shown in Figure 2.5. In order to obtain data with a constant 5-minute sampling time two imputation methods were implemented: Gaussian process regression (GPR) and linear interpolation. The initial method entailed training and fitting a Gaussian process (GP) to the CGM data. Despite rigorous effort considering numerous kernel and hyperparameter combinations, this approach failed to provide viable interpolations and was consequently abandoned (see A.3 for further discussion). To avoid look-ahead bias, the inputs and targets were re-sampled separately. To circumvent look-ahead bias, inputs and targets were re-sampled independently. The interpolated CGM measurements were then re-sampled at 5-minute intervals to ensure consistent CGM value count for each sample.

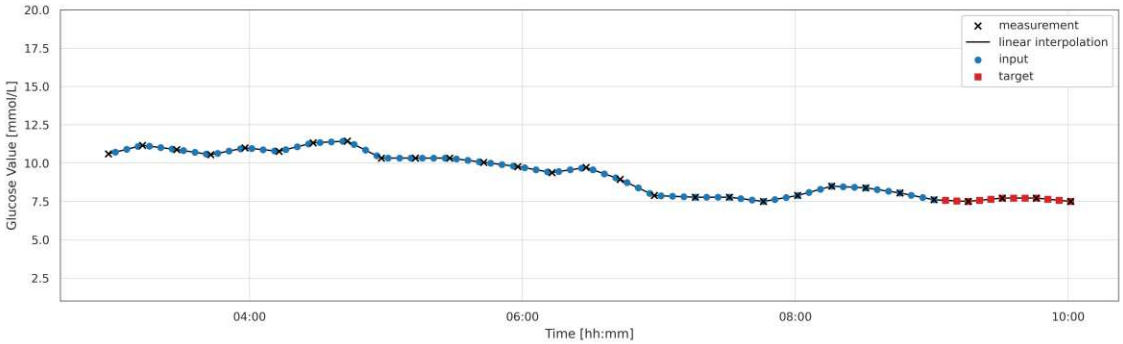


Figure 2.5.: An example of linear interpolation of measurements and re-sampling of input and target data. The measurements, originating from a FreeStyle Libre sensor, are sampled approximately every 15 minutes, while inputs and targets are re-sampled precisely every 5 minutes. To prevent look-ahead bias, the last input point always aligns with a measurement.

2.2. Probabilistic CGM forecasting

2.2.1. Problem statement

Based on previous CGM forecasting works surveyed in Oviedo et al. [6] and Woldaregay et al. [7] we aim to forecast CGM values 30, 60 and 120 minutes into the future in 5-minute increments based on the CGM values of the last 6 hours. With the input duration of 6 hours and prediction horizons (PHs) of 30, 60, and 120 minutes, along with 5-minute interval sampling, the input data falls into the set $\mathcal{X} \subseteq \mathbb{R}^{72}$ while the output resides in the set $\mathcal{Y} \subseteq \mathbb{R}^h$, with target lengths given by $h \in 6, 12, 24$. For a given input-target-pair $(x, y) \in \mathcal{X} \times \mathcal{Y}$ a probabilistic forecasting the model f maps the input $x \in \mathcal{X}$ to a probability distribution $f(x) = \mathbb{P}$ over $\mathcal{Y}$. The objective is to identify a model f such that for every $x \in \mathcal{X}$ the corresponding target y is likely to be sampled from the probability distribution $f(x) = \mathbb{P}$ to which x is mapped. In the context of CGM forecasting, it has been demonstrated that independent normal distributions are efficacious for this purpose [20, 40].

Since $f(x) = \mathbb{P} = \mathcal{N}(\mu, \Sigma)$, with $\mu, \sigma^2 \in \mathbb{R}^h$ and $\Sigma = \text{diag}(\sigma^2)$, is uniquely defined by μ and σ^2 we can identify the space of all independent normal distributions with $\mathbb{R}^h \times \mathbb{R}^h_+$, the space of possible values for μ and σ^2. As a result, the models under consideration adopt the form

$$f \colon \mathcal{X} \to \mathbb{R}^h \times \mathbb{R}^h_+ \tag{2.1a}$$

$$x \mapsto (\mu, \sigma^2) \tag{2.1b}$$

A loss function $\mathcal{L}_y(\mu, \sigma^2)$ is utilized to measure the unlikelihood of $y \in \mathcal{Y}$ being drawn from $\mathcal{N}(\mu, , \text{diag}(\sigma^2))$. The chosen model $f \in \mathcal{F}$ resides within a model space $\mathcal{F}$. The potential model spaces $\mathcal{F}$ are discussed in 2.2.2 and 2.4.1, with the loss $\mathcal{L}$ being represented by the sum of Gaussian NLLs, elaborated on in 2.2.3.

With the aforementioned notation, the task of probabilistic CGM forecasting can be stated as a special case of empirical risk minimization (ERM), where we seek to find an optimal model within the given model space, that minimizes the mean loss for the provided dataset:

Definition 2.2.1 (Empirical risk minimization)**.** For a given training set with $n \in \mathbb{N}$ samples $\mathcal{D} \subseteq \mathcal{X} \times \mathcal{Y}$ and a model space $\mathcal{F}$, find a model $f^* \in \mathcal{F}$ that minimizes the *empirical risk* $R_{emp}(f) := \frac{1}{n} \sum_{(x,y)\in\mathcal{D}} \mathcal{L}_y(f(x))$ if a minimum exists, i.e.,

$$f^* = \arg\min_{f \in \mathcal{F}} R_{emp}(f).$$

2.2.2. Baseline models

Tackling the task of probabilistic CGM forecasting, we assume a general model f of the form (2.1). This section details three models: the t_{-1} model, an LSTM model, and a transformer model adapted for time series forecasting. The LSTM and transformer architectures represent model spaces in the sense of 2.2.1, since they are neural network architectures and as such they come with tunable parameters, where every set of given parameters constitutes a model within the respective model space.

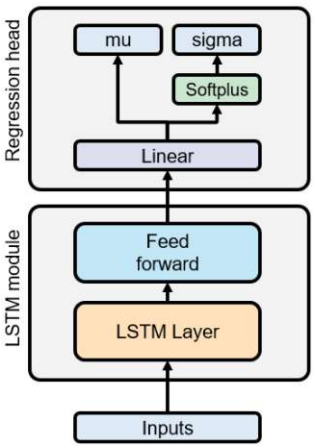
Baseline t_{-1}. The model t_{-1} is a simple model that sets the stage for evaluating the improvements brought in by the more complex models. It is motivated by the naive forecast which is used as a baseline in non-probabilistic forecasting [41], but adapted to incorporate a prediction for the uncertainty based on the uncertainty of the naive forecast on the training data. Let $\mathcal{D}_{train} \subseteq \mathcal{D} \subseteq \mathcal{X} \times \mathcal{Y}$ denote a training set and for an input $x \in \mathcal{X}$ let $t_{-1}(x) = (\mu(x), \sigma^2)$ be the baseline model's output. At each prediction time $i = 1, \ldots, h$ the mean prediction $\mu_i(x)$ is the last known input value and the variance is the variance over the train set $\mathcal{D}_{test}$, i.e.,

$$\mu(x)_i := x_{72}$$

$$\sigma_i^2 := \frac{1}{|\mathcal{D}|} \sum_{(v,w)\in\mathcal{D}_{train}} (\mu(v)_i - w_i)^2$$

for $i = 1, \ldots, h$.

Long short-term memory (LSTM) model. The original LSTM architecture was developed by Hochreiter and Schmidhuber [42] to learn short and long-term dependencies in time series data. This property was the choice of an LSTM model for the task of CGM forecasting. The LSTM architecture we use was developed by Martinsson et al. [20], where a grid search over the LSTM hyperparameter space was performed to derive an optimal architecture in terms of RMSE on the Ohio T1DM dataset [43] for probabilistic CGM forecasting. The architecture is presented in Table 2.6. It incorporates a single LSTM layer with a hidden state size of 256. The output of the LSTM layer is fed through a feed-forward network with two fully-connected layers. The two layers' output sizes are 512 as well as 256, respectively and each is followed by a rectified linear unit (ReLU) activation. Finally, the resulting feature vector $\tilde{x} \in \mathbb{R}^{256}$ is fed into the regression head. In the regression head the feature vector $\tilde{x}$ is duplicated and both copies are linearly transformed to match the target length h, where $h \in \{6, 12, 24\}$ represents the PH (one prediction every 5 minutes), resulting in two vectors $o_1, o_2 \in \mathbb{R}^h$. The first vector's entries are the predictions of the means $\mu = o_1$, whereas to the second vector o_2 a softmax function $\text{softmax}(x)_i := e^{x_i} \,/\, \left(\sum_{j=1}^{h} e^{x_j}\right)$ is applied to produce the non-negative predictions of the variances $\sigma = \text{softmax}(o_2)$. This architecture is used as a standalone model and as a feature extractor (FE) (without the regression head) for the more advanced ensemble and GDU models discussed in 2.4.1.



LSTM model architecture	
Layer type	Output shape
LSTM	(256)
Linear	(512)
ReLU	(512)
Linear	(256)
ReLU	(256)
Linear	(2, h)
Softplus	(2, h)

Figure 2.6.: The LSTM model's architecture for probabilistic CGM forecasting. It was developed by Martinsson et al. [20] through hyperparameter optimization and it showed state-of-the-art performance while being computationally inexpensive to train.

Transformer model (TM). The adapted transformer architecture we use was first introduced by Li et al. [29] in a successful attempt to make the classic transformer architecture [26] less prone to anomalies and thus a viable option for time series forecasting. We utilize the convolutional self-attention presented in Li et al. [29]. The transformer architecture was modified for probabilistic CGM forecasting and the modified version is depicted in Figure 2.7. The main modification to the TM as presented in Li et al. [29] is that the output of the model is transformed into two vectors μ, σ instead of one and that the sparse propagation of information is not utilized in this work, because of the shorter input length of our data. The architecture consists of three parts. First, the inputs are embedded using learnable position embeddings, since the following attention module

Table 2.2.: The transformer model's architecture for probabilistic CGM forecasting. The architecture is based on the transformer model (TM) developed by Li et al. [29].

TM architecture	
Layer type	Output shape
Concatenation	(72, 9)
1D Convolution	(72, 9), (72, 9), (72, 9)
Multiplication	(72, 72), (72, 9)
Softmax	(72, 72), (72, 9)
Multiplication	(72, 9)
Linear	(72, 9)
Normalization	(72, 9)
Linear	(72, 36)
ReLU	(72, 36)
Linear	(72, 9)
ReLU	(72, 9)
Normalization	(72, 9)
Linear	(2, h)
Softplus	(2, h)

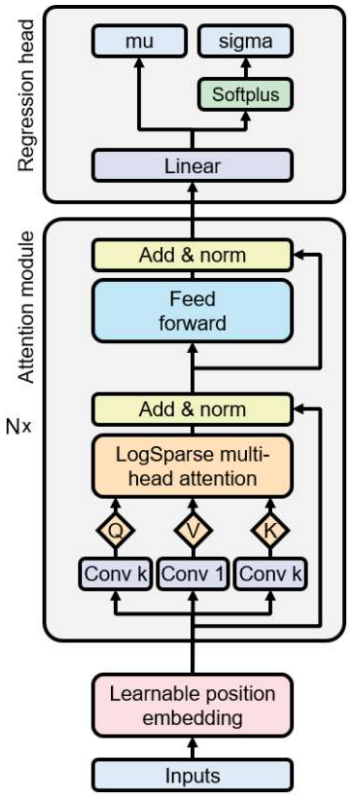


Figure 2.7.: The transformer model architecture for probabilistic CGM forecasting. The architecture consists of three main parts: The input time series is processed by the learnable position embedding before it runs iteratively through the convolutional attention block. Lastly, the outputs are transformed by the regression head to fulfill the requirements of CGM forecasting.

of the architecture is position-insensitive. The learnable position embeddings are learnable parameters $W^{PE} \in \mathbb{R}^{n \times p}$ of length p for every one of the n input positions, that are concatenated to the input entries $x = (x_1, \ldots, x_n)^T \in \mathbb{R}^n$ to capture the notion of each input's position in the time series [44], resulting in the embedding matrix $X \in \mathbb{R}^{n \times (p+1)}$. Following is the attention module stacked N times. It utilizes multi-head convolutional self-attention with H head, which is briefly described here. For every head $i \in \{1, \ldots, H\}$, the module's input matrix X is simultaneously convolved along the temporal dimension with kernel size k, stride s, $p+1$ channels, and the necessary padding to obtain the attention head's query $Q_i \in \mathbb{R}^{n \times (p+1)}$ and key $K_i \in \mathbb{R}^{n \times (p+1)}$ matrices. The value matrix $V_i = W_i^V X \in \mathbb{R}^{n \times (p+1)}$ is computed as the matrix product between the learnable parameters $W_i^V \in \mathbb{R}^{n \times n}$ and X. These linear operations are followed by the scaled dot-product attention

$$O_i = \text{Attention}(Q_i, K_i, V_i) = \text{softmax}\left(\frac{Q_i K_i^T}{\sqrt{(p+1)}} \cdot M\right) V_i,$$

with $M \in \mathbb{R}^{n \times n}$ a lower triangular mask matrix that filters out rightward attention to avoid future information leakage. The attention matrices from every head $O_1, \ldots, O_H$ are concatenated and projected to the size of the attention module's input $X \in \mathbb{R}^{n \times (p+1)}$

to produce the attention. The module's input X is added to the attention and layer normalization is performed [45]. The normalized attention is fed through a 2-layer perceptron with output dimensions $(n, 4(p+1))$ and $(n, (p+1))$ which are separated by a ReLU activation. To the perceptron's output, the normalized attention is added before another layer normalization concludes the attention module. Lastly, the same regression head as for the LSTM model is applied.

The novelty from the TM used in this work is convolutional self-attention, which takes a convolution for the query and key in the attention mechanism as described and depicted in 2.7. Whether an observed point is an anomaly, change point or part of a pattern is highly dependent on its surrounding context, which is captured through the convolution [29]. The application for CGM forecasting is further motivated by the superior performance of TM over other state-of-the-art time series forecasting algorithms on a variety of datasets. The selection of hyperparameters is based on the experiments and ablation study presented in the paper. The input length of $n = 72$ stems from the given CGM data and for the dimension of position embeddings $p = 8$ is set. The convolution size is $k = 4$ with stride $s = 1$. The number of attention heads is $H = 1$ and the attention module is stacked once, $N = 1$.

2.2.3. Evaluation measures

Given a training and test split of the dataset $\mathcal{D}$ containing input-target CGM samples, each model's efficacy was evaluated on the test set using several metrics to ascertain both its clinical and technical significance.

Root-mean-square error (RMSE). The RMSE is the predominant metric employed in CGM prediction as it provides a straightforward and interpretable measure of model performance [7, 20, 40, 46, 47]. RMSE quantifies the error between targets and predictions as for two vectors $y, \hat{y} \in \mathbb{R}^h$ it is defined as $\mathcal{L}(y, \hat{y}) := \sqrt{\sum_{i=1}^{h}(y_i - \hat{y}_i)^2}$. In the case of probabilistic forecasting with a multivariate Gaussian distribution of the form $\mathcal{N}(\mu, \Sigma)$ as a model's output, the prediction of y is defined as the mean vector of the distribution $\hat{y} := \mu \in \mathbb{R}^h$. We used the RMSE to evaluate our trained models' technical performances on the hold out test data. Note, that the RMSE does not make use of the predicted variance σ^2. We report it since it serves as an intuitive metric which is regularly used in CGM prediction research [18, 19, 23].

Negative log-likelihood (NLL). The Gaussian negative log-likelihood constitutes a measure of how unlikely it is for a given vector $y \in \mathbb{R}^h$ to be sampled from a Gaussian distribution $\mathcal{N}(\mu, \mathrm{diag}(\sigma^2))$ with $(\mu, \sigma^2) \in \mathbb{R}^h \times \mathbb{R}^h_+$. The NLL loss of a model's prediction (μ, σ^2) w.r.t. a given target vector y is derived from the NLL $\mathcal{L}_y(\mu, \sigma^2)$ by dropping constants and introducing a variance clamp $\epsilon = 1e^{-6}$ for numerical stability. The NLL loss is defined

as

$$\mathcal{L}_y(\mu, \sigma^2) := \frac{1}{2} \sum_{i=1}^{h} \log\left(\max(\sigma_i^2, \epsilon)\right) + \frac{(\mu_i - y_i)^2}{\max(\sigma_i^2, \epsilon)} \tag{2.2}$$

and we used it for model training, validation as well as testing.

Error grid analysis. CGM devices are prone to inaccuracies and to gauge the clinical precision of these devices, error grid analyses are performed. in which the true glucose values are plotted against the measured CGM values. his process entails plotting the actual glucose values against the CGM measurements, with each measurement falling into one of several predefined areas on the two-dimensional plane, which signify the clinical significance of the measurement error.

Introduced in Clarke et al. [30], the CEG outlines five zones as shown in Figure 2.8a:

A clinically accurate,

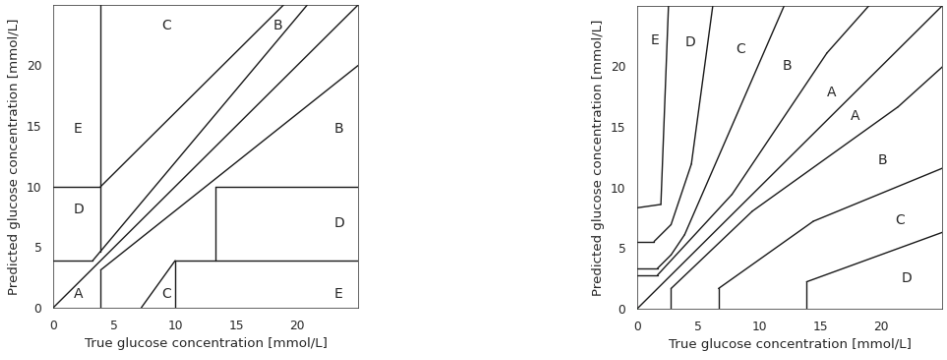
B incorrect, but benign treatment might be given,

C an over corrective treatment might be given,

D an error was not detected,

E severe erroneous treatment might be given.

The PEG shares the same zone labels as the CEG, but the zone boundaries were redefined to more accurately reflect the medical outcomes of measurement errors [31], as depicted in Figure 2.8b. Conducting an error grid analysis is a standard practice in CGM prediction, measuring the percentages of predictions falling into zones *A*-*E* to yield a numerical assessment of the clinical severity of errors made by the models [6, 7]. We evaluated the clinical performance of the trained models on the reserved test data using the percentages per PEG zone, and based on the PEG, we developed a novel PEG loss to enhance the optimization towards clinically relevant CGM prediction models.



(a) The CEG regions.

(b) The improved PEG regions.

Figure 2.8.: The Clarke error grid (CEG) and Parkes error grid (PEG) regions, describing the clinical severity of errors in CGM measurements.

2.3. Development of the Parkes error grid loss for clinically relevant predictions

On a non-trivial task, every model makes erroneous predictions. Depending on the use case, but certainly true when the model's outputs inform medical interventions, the errors the model makes, can be categorized by severance. Therefore it is common practice to employ loss functions, which take into consideration the severance of a prediction's error [15, 48]. This approach holds particular relevance for CGM forecasting, as inaccurate prediction of high CGM values during a hypoglycemic episode could prompt interventions that further decrease blood glucose levels. Since by definition the NLL does not differentiate between the direction of an error, there is a need to identify a loss function that does.

2.3.1. Introduction of the Parkes error grid loss

In pursuit of clinically relevant models, some research [15] has utilized the CEG to develop a loss function for model training, thus quantifying the clinical implications of a model's predictions. Given that the CEG is now outdated [31], we opt for the more applicable PEG as the foundation for our novel loss function formulation.

In search of a loss function that is interpretable and compatible with the setting of probabilistic CGM forecasting, a loss of the form

$$\mathcal{L}(y, (\mu, \sigma)) = \mathcal{L}_y(\mu, \sigma^2) + \lambda_{PEG}\hat{\mathcal{L}}_{PEG}(y, \mu)$$

is desired, were $\mathcal{L}_y$ is the NLL loss (2.2), $\lambda > 0$ is a weight and

$$\hat{\mathcal{L}}_{PEG}(y, \mu) := \frac{1}{h}\sum_{i=1}^{h} \mathcal{L}_{PEG}(y_i, \mu_i)$$

is an additional term based on the PEG. We derive desired properties of the PEG loss $\mathcal{L}_{PEG}$ from the desired properties of loss functions in regression tasks [49]. The desired properties are the following:

(i) It takes its global minimum 0 on the diagonal $\mu = y$, i.e., $\mathcal{L}_{PEG}(y, y) = 0$ for all $y \in \mathbb{R}_+$ and $\mathcal{L}_{PEG}(y, \mu) \geq 0$ for all $y, \mu \in \mathbb{R}_+$,

(ii) it grows in value with the clinical danger of the prediction μ,

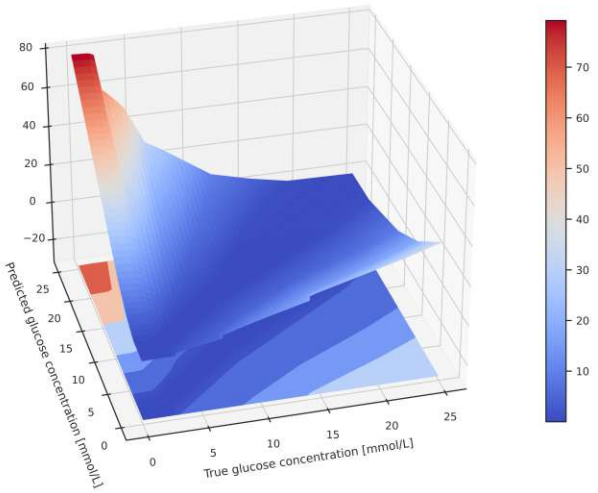
(iii) it is continuous and piecewise differentiable in the direction of the prediction μ to ensure that variations of gradient descent are applicable,

(iv) its partial derivative with respect to μ, namely $\frac{\partial \mathcal{L}_{PEG}}{\partial \mu}$ grows in absolute value with the clinical danger of a prediction to result in fast convergence during model training,

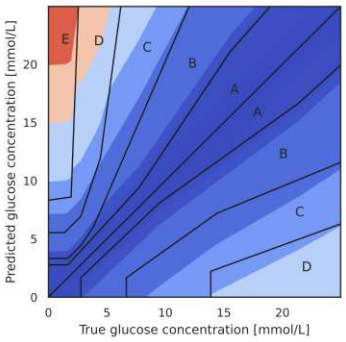
(v) the calculation of the loss and its partial derivative with respect to μ should be computationally efficient.

We denote with A_+, B_+, C_+, D_+, E_+ the subareas above and with A_-, B_-, C_-, D_- the subareas below the diagonal. Additionally, we define the slopes $0 \leq s_A \leq s_B \leq s_C \leq s_D \leq s_E$.

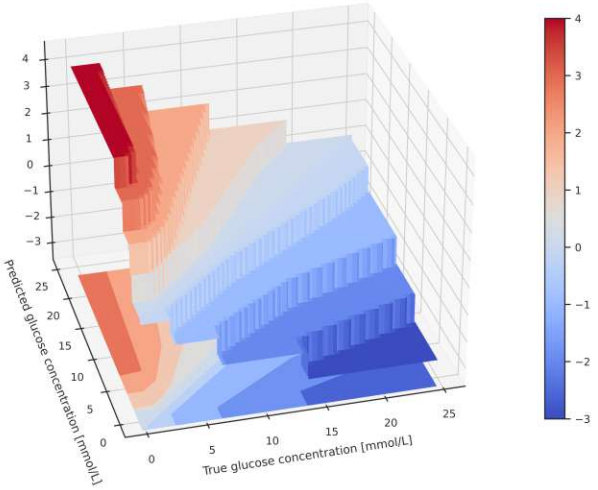
Candidates for the PEG loss $\mathcal{L}_{PEG}$ are the functions that are continuous in μ-direction and are zero on the diagonal. The loss is uniquely defined by enforcing that for every point (y, μ) and the subarea it is an element of $X \in \{A_\pm, B_\pm, C_\pm, D_\pm, E_+\}$ the slope in μ-direction should be $\pm s_X$. The visualization in Figure 2.9 outlines the alignment with the PEG and visualizes the desired properties. By definition this continuous and piecewise linear loss function $\mathcal{L}_{PEG}$ satisfies the properties (i) - (iv). The formulae for the loss and its partial derivative as well as property (v) are detailed in the following.



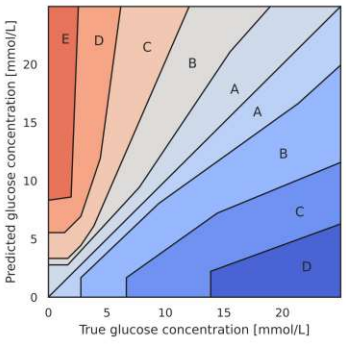
(a) Surface plot of the PEG loss $\mathcal{L}_{PEG}$



(b) Contour plot of the PEG loss $\mathcal{L}_{PEG}$ in alignment with the PEG



(c) Surface plot of $\frac{\partial \mathcal{L}_{PEG}}{\partial \mu}$



(d) Contour plot of $\frac{\partial \mathcal{L}_{PEG}}{\partial \mu}$ in alignment with the PEG

Figure 2.9.: The visualization of the PEG loss 2.9a, its partial derivative 2.9c, and their alignment with the PEG 2.9b, 2.9d. The PEG loss was defined, such that itself and its partial derivative take large absolute values in regions of the PEG that indicate clinically relevant mistakes. These two properties shall ensure faster convergence to models that make less clinically significant errors.

In order to derive a formulaic expression for the PEG loss, the definition of the borders of the subareas $A_\pm, B_\pm, C_\pm, D_\pm, E_+$ are necessary:[1]

$$b_{A_+}(y) = b_{A_-}(y) = y$$

$$b_{B_+}(y) = \begin{cases} 2.77, & y \leq 1.66 \\ 1.09y + 0.96, & 1.66 < y \leq 7.77 \\ 1.5y - 2.22, & 7.77 < y \leq 15.54 \\ 1.13y + 3.48, & 15.54 < y \end{cases} \quad b_{B_-}(y) = \begin{cases} 0, & y \leq 2.77 \\ 0.96y - 0.99, & 2.77 < y \leq 9.43 \\ 0.72y + 1.24, & 9.43 < y \leq 21.36 \\ 0.91y - 2.77, & 21.36 < y \end{cases}$$

$$b_{C_+}(y) = \begin{cases} 3.33, & y \leq 1.66 \\ y + 1.66, & 1.66 < y \leq 2.77 \\ 1.5y + 0.28, & 2.77 < y \leq 3.88 \\ 2.31y - 2.89, & 3.88 < y \end{cases} \quad b_{C_-}(y) = \begin{cases} 0, & y \leq 6.66 \\ 0.71y - 3.09, & 6.66 < y \leq 14.43 \\ 0.41y + 1.24, & 14.43 < y \end{cases}$$

$$b_{D_+}(y) = \begin{cases} 5.55, & y \leq 1.39 \\ y + 4.16, & 1.39 < y \leq 2.77 \\ 3y - 1.38, & 2.77 < y \leq 4.44 \\ 7.44y - 21.12, & 4.44 < y \end{cases} \quad b_{D_-}(y) = \begin{cases} 0, & y \leq 13.87 \\ 0.37y - 2.87, & 13.87 < y \end{cases}$$

$$b_{E_+}(y) = \begin{cases} 0.14y + 8.32, & y \leq 1.94 \\ 26.33y - 42.54, & 1.94 < y \end{cases} \quad b_{E_-}(y) = 0.$$

The height of a subarea $X \in \{A_\pm, B_\pm, C_\pm, D_\pm\}$ at a given glucose value y is then expressible by

$$h_X(y) := |b_{s(X)}(y) - b_X(y)|,$$

where $s(X)$ denotes the area that is following area X in terms of clinical severity, e.g., $s(A) = B$ or $s(D) = E$. Consistently, $M(X)$ denotes the set containing all areas that are clinically milder than area X, e.g., $M(E) = \{A, B, C, D\}$ or $M(A) = \emptyset$. Using this notation, the PEG loss of a point (y, μ) in subarea $X_\pm$ can be written as

$$\mathcal{L}_{PEG}(y, \mu) = \pm s_X(\mu - y) - \sum_{Y \in M(X)} (s_X - s_Y) h_{Y_\pm}(y).$$

Therefore, the partial derivative w.r.t μ is easily computable as

$$\frac{\partial \mathcal{L}_{PEG}}{\partial \mu}(y, \mu) = \pm s_X,$$

which allows the fast computation of the loss and its gradient for variations of gradient descent during model training, satisfying property (v).

For model training, the hyperparameter λ is initially set to 1 and in the ablation study in Section 3.3, the resulting models' sensitivity to the parameter is shown.

[1] Note that the coefficients in the definition are calculated from the provided coordinates of the lines defining the areas of the PEG [31].

2.3.2. Training and evaluation procedure

In order to examine the advantages of applying the PEG loss for model training, both LSTM and TM, as described in Section 2.2.2, are trained and assessed with and without the PEG loss term.

To accurately evaluate the models' performance on unseen data, we employed individual-based 5-fold cross-validation. Each individual in the dataset was assigned a unique ID, barring 20 individuals whose data files were given two separate IDs due to an unnoticed error from the data provider. The data was divided according to the individuals' IDs, ensuring each fold contained an equal number of samples, and that data from a single ID was present only in one fold. The dataset $\mathcal{D}$, comprising input-target CGM samples, was partitioned into five folds $\mathcal{D}_1, \ldots, \mathcal{D}_5$ of equal size, such that each individual's data is encapsulated in just one of the folds. For every test fold $\mathcal{D}_{test} := \mathcal{D}_i$ the remaining data $\bigcup_{j \neq i} \mathcal{D}_i$ is randomly split into training $\mathcal{D}_{train}$ and validation data $\mathcal{D}_{val}$ following an 80/20-split. The splitting of training and validation data conformingly took into account each individual's data.

The same hyperparameters for model training were set to achieve comparability between the results of [20] and us. The batch size was fixed at 1024 and the initial learning rate was set to 10^{-1}. We used an exponentially decaying learning rate with a multiplicative decay rate of $\gamma = 0.999$, in conjunction with an Adam optimizer. The maximum number of training epochs was set to 10000 and early stopping was utilized, halting model training if the loss on the validation data $\mathcal{D}_{val}$ did not improve for 200 epochs.

The performance of the trained models was assessed on the test fold $\mathcal{D}_{test}$ using the RMSE, NLL, and the PEG metrics. For every test sample $(x, y) \in \mathcal{D}_{test}$ the respective metric was calculated for every time point within the PH. The metrics of each time point of a sample were averaged before calculating the average over all test samples. Finally, the calculated metrics of every testing set $\mathcal{D}_{test} \in \{\mathcal{D}_1, \ldots, \mathcal{D}_5\}$ were averaged to obtain the final values reported in 3.

2.4. Domain generalization in CGM forecasting with gated-domain-units

Visual, numerical, and medical evidence depicted in Figure 2.1, Table 2.1, and provided by the staff at *Inselspital* suggest that the distribution of CGM trajectories changes over time due to variations in influencing factors, such as diabetes treatments, devices, and insulin types. These distributional shifts may diminish the performance of the developed forecasting models. To counteract the adverse effects of distribution shifts, we measured and improved the DG capabilities of the models.

In DG, we consider M source domains $\mathcal{D}^{train} = \{\mathcal{D}^m\}_{m=1}^M$ with $\mathcal{D}^m = \{(x_i^m, y_j^m)_{i=1}^{n_m}\} \sim \mathbb{P}_m$ for $m = 1, \ldots, M$ and a target domain $\mathcal{D}^{test} \sim \mathbb{P}_{test}$, such that the distributions are in pairs different, i.e., $\mathbb{P}_j \neq \mathbb{P}_k$ for $j \neq k$. The objective is to obtain a model from the M source domains $\mathcal{D}^{train}$ that minimizes the loss on the test domain $\mathcal{D}^{test}$.

The setting of DG by definition violates the i.i.d assumption, which is crucial to statistical learning algorithms [8]. This violation can lead to diminished model performance on the test domain, as evidenced in the literature [9, 50]. Notably, the complexities of DG have been under rigorous academic scrutiny for over a decade [9]. In healthcare domains, the challenges posed by these distributional shifts, and the underlying factors that contribute to them, have been meticulously examined [11]. Given the indications that novel treatment modalities could be driving these shifts in our dataset, we are poised to implement a novel solution: the GDUs approach [33].

2.4.1. Introduction of gated-domain-units

For the DG experiments, the LSTM model served as the baseline and is henceforth referred to as the ERM single in the context of DG. The models addressing DG presented in this section are two ensemble models that utilize the LSTM module as a feature extractor.

Empirical risk minimization (ERM) ensemble. Ensemble learning is a common technique to increase the OOD performance [9] and it yields maximal comparability with the GDU model due to its similar structure. The ERM ensemble model is built on top of the LSTM module, which is used as a FE, see Figure 2.10. Let the feature vector $\tilde{x} \in \mathbb{R}^{256}$ denote the output of the FE, then instead of applying a linear layer to obtain μ and σ, the feature vector is copied $M = 9$ times and fed through M linear layers in parallel to obtain $\mu^1, \ldots, \mu^M$ and $\sigma^1, \ldots, \sigma^M$. The final output of the ensemble model is given by the pointwise average of the M outputs, i.e., $\mu = \frac{1}{M} \sum_{j=1}^{M} \mu^j$ and $\sigma = \frac{1}{M} \sum_{j=1}^{M} \sigma^j$.

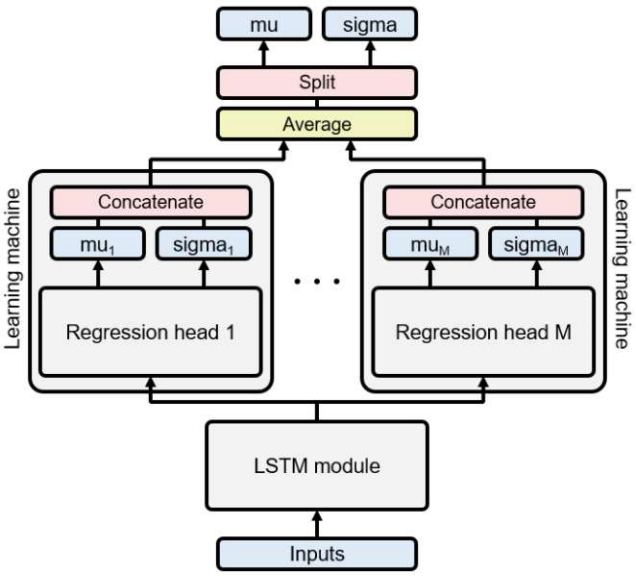


Figure 2.10.: The ERM ensemble architecture for probabilistic CGM forecasting. The architecture uses an LSTM module as a feature extractor. The extracted features are passed through $M = 9$ regression heads in parallel and the M outputs are averaged to obtain the final predictions for μ and σ.

Gated-domain-units (GDU) model. GDUs were introduced by Föll et al. [33] and are based on the assumption that real-world distributions are composed of elementary distributions that remain invariant across different domains. This assumed consistency across domains can facilitate knowledge transfer to previously unseen domains (e.g., treatment strategies, hospitals, countries, etc.). The utilization of GDUs has shown substantial improvements in performance across numerous DG tasks when compared to the ERM ensemble architecture and other baselines.

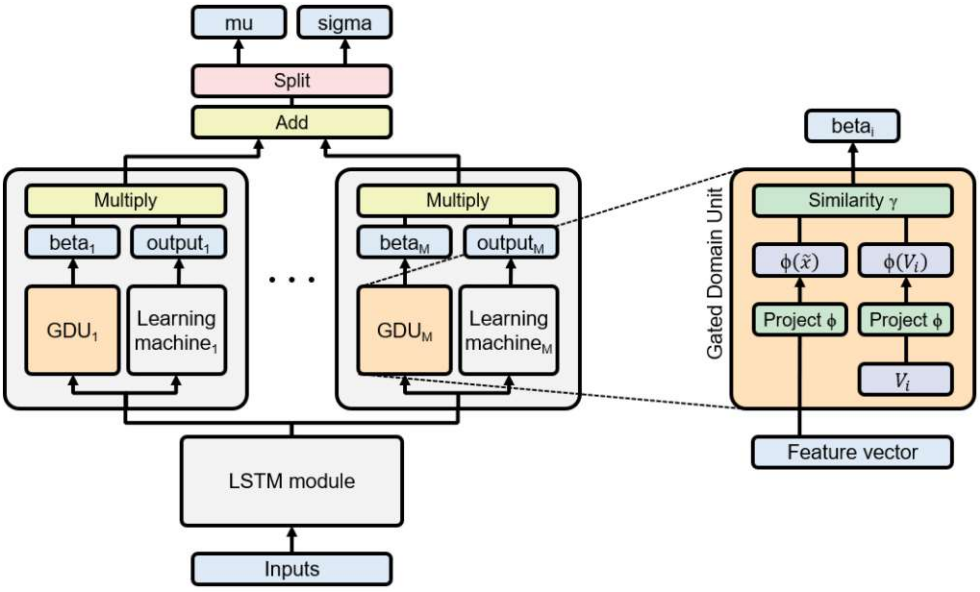


Figure 2.11.: The gated-domain-units (GDU) model architecture for probabilistic CGM forecasting. The architecture uses an LSTM module as a feature extractor. The extracted features are passed through M regression heads and their respective GDUs in parallel to obtain M outputs (μ_i, σ_i) with their corresponding weights β_i. The weighted averages are the final predictions for μ and σ.

Structurally, the GDU model echoes the ERM ensemble model, with the key distinction being that the outputs $\mu^1, \ldots, \mu^M$ and $\sigma^1, \ldots, \sigma^M$ are assigned weights during averaging,

$$\mu = \frac{1}{M} \sum_{j=1}^{M} \beta_j \mu^j,$$

with weights $\beta_1, \ldots, \beta_M \in [0, 1]$ and $\sum_{j=1}^{M} \beta_j = 1$. Each weight β_j quantifies the similarity between the feature vector $\tilde{x} \in \mathbb{R}^{256}$ and the elementary domain's distribution $\mathbb{P}_j$ which is approximated by the so-called elementary domain basis $V_j = (v_1^j, \ldots, v_N^j) \in \mathbb{R}^{256 \times N}$ of a learnable N-dimensional subspace of the feature space $\tilde{\mathcal{X}} \subset \mathbb{R}^{256}$.

The interpretation of the learnable bases $V_1, \ldots, V_M$ and calculation of the weights $\beta_1, \ldots, \beta_M$ are detailed as follows: The overall objective is to identify elementary domains and quantify similarities between them and samples. To this end let $\mathcal{H}$ be a reproducing kernel Hilbert space (RKHS) of real-valued functions on the feature space $\tilde{\mathcal{X}}$ with a reproducing kernel $k : \tilde{\mathcal{X}} \times \tilde{\mathcal{X}} \to \mathbb{R}$ [51]. Given the samples $\tilde{x}^1, \ldots, \tilde{x}^n$ from an elementary domain's distribution $\mathbb{P}$, the distribution can be approximated in the RKHS as $\phi(\mathbb{P}) \approx \frac{1}{n} \sum_{i=1}^{n} k(\tilde{x}^i, \cdot)$. To overcome the non-accessibility of samples from the elementary distributions, a set of proxy vectors $v_1, \ldots, v_N$ is sought to replace the samples $\tilde{x}^1, \ldots, \tilde{x}^n$. This elementary domain basis is collected into a matrix $V = (v_1, \ldots, v_N)$ and used to approximate $\phi(\mathbb{P}) \approx \phi(V) = \frac{1}{n} \sum_{i=1}^{n} k(v_i, \cdot)$.

For a given sample $\tilde{x}$ the sought-after weights $\beta_1, \ldots, \beta_M$ are defined as the similarity between the sample and the respective elementary domain basis $V_1, \ldots, V_M$. The computability of

products in the RKHS, as detailed in A.4, enables the calculation of similarity measures to calculate the sought-after weights $\beta_1, \ldots, \beta_M$. Two similarity measures are considered in the following: the cosine similarity (CS) $H_{CS}(u, v) = \frac{\langle u,v\rangle_{\mathcal{H}}}{\|u\|_{\mathcal{H}}\|v\|_{\mathcal{H}}}$ is an angle-based similarity measure, whereas the negative maximal mean discrepancy (MMD) $H_{MMD}(u, v) = -\|u-v\|_{\mathcal{H}}$ is distance-based. Once the similarities between a sample and the elementary domain bases is calculated, a softmax is applied to the resulting values to ensure the resulting weights $\beta_1, \ldots, \beta_M$ are non-negative and sum to 1, i.e.,

$$\beta_j = \gamma\left(\phi(\tilde{x}), \phi(V_j)\right) = \frac{\exp\left(\kappa H\left(\phi(\tilde{x}), \phi(V_j)\right)\right)}{\sum_{k=1}^{M} \exp\left(\kappa H\left(\phi(\tilde{x}), \phi(V_k)\right)\right)}.$$

During training of a GDU-model g, two additional loss terms with corresponding coefficients are introduced,

$$\mathcal{L}(y, (\mu, \sigma), g) = \mathcal{L}_y(\mu, \sigma^2) + \lambda_{OLS}\Omega_D^{OLS}(\|g\|_{\mathcal{H}}) + \lambda_{L_1}\Omega_D^{L_1}(\|\gamma\|), \quad \lambda_{OLS}, \lambda_{L_1} \geq 0. \tag{2.3}$$

The term $\Omega_D^{OLS}(\|g\|_{\mathcal{H}}) := \|\phi(\tilde{x}) - \sum_{j=1}^{M} \beta_j \phi(V_j)\|_{\mathcal{H}}^2$ ensures that the distances between the feature mapping $\phi(\tilde{x})$ and the associated representations $\sum_{j=1}^{M} \beta_j \phi(V_j)$ are minimized and therefore, the elementary domain bases V_j are able to represent the elementary domains $\mathbb{P}_j$ in the RKHS. Lastly, sparsity in the weight vector $\beta = (\beta_j)_{j=1}^{M} = (\gamma(\phi(\tilde{x}), \phi(V_j)))_{j=1}^{M}$ is enforced by applying the L_1-norm in the term $\Omega_D^{L_1}(\|\gamma\|) := \|\beta\|_1$. A visualization of the GDU-model is given in Figure 2.11.

As a kernel k we chose the radial basis function (RBF) kernel $k(x, y) = \exp\left(-\|x - y\|_2^2/2\sigma^2\right)$ and set the parameter σ^2 with the median heuristic as in Föll et al. [33] and Muandet et al. [51], i.e., $\sigma^2 = \text{median}\{\|\tilde{x} - \tilde{y}\|_2^2 : x, y \in \mathcal{D}_{train}\}$. The number $M = 9$ of learning machines and GDUs, respectively was set as proposed in Föll et al. [33] by clustering the output of a feature extractor with the k-means algorithm and selecting the number of clusters that maximizes the Calinski-Harabasz score. Lastly, the dimension N of the elementary domain bases was set to $N = 10$ as in Föll et al. [33].

In the experiments conducted, we utilized two distinct modes of training: one being fine tuning (FT), wherein the feature extractor is a pre-trained model with its weights remaining fixed, and the other being end-to-end training (E2E), in which the feature extractor is simultaneously trained with the GDUs, without any prior training.

2.4.2. Training and evaluation procedure

To measure and augment the DG performance of the LSTM model, we carried out a series of experiments as delineated in this subsection. Based on the guidance from the Inselspital's medical team, the available CGM dataset $\mathcal{D} \subset \mathbb{R}^{72} \times \mathbb{R}^h$ was partitioned according to the treatment types into four separate domains: basal insulin only $\mathcal{D}_{BI}$, multiple daily injections $\mathcal{D}_{MDI}$, continuous subcutaneous insulin infusion $\mathcal{D}_{CSII}$, and artificial pancreas $\mathcal{D}_{AP}$. For each domain $\mathcal{D}_{test} \in \{\mathcal{D}_{BI}, \mathcal{D}_{MDI}, \mathcal{D}_{CSII}, \mathcal{D}_{AP}\}$, model training was conducted using only the remaining three domains $\mathcal{D}_{train} = \mathcal{D} \setminus \mathcal{D}_{test}$. The models were subsequently evaluated on the test domain $\mathcal{D}_{test}$.

The evaluation metrics incorporated for these assessments include RMSE, NLL, and the percentage of measurements within specific areas of the PEG. To mitigate the effects of randomness, the entire process of training and evaluation was replicated five times, each instance employing a unique random seed for model initialization. The derived metrics' mean and standard deviation are documented in 3.2.

Maintaining consistency with the model training hyperparameters from 2.3.2 allowed for comparability and negated the requirement for hyperparameter tuning, thus saving computational resources. A batch size of 1024 was used alongside the loss function detailed in 2.3. The training also involved the application of the Adam optimizer, coupled with a learning rate experiencing exponential decay at a multiplicative rate of $\gamma = 0.999$. The maximum epoch limit for the training was set at 10000, with the incorporation of early stopping to halt the training process if there was no improvement in the validation data loss for 200 consecutive epochs.

2.5. Implementation overview

The investigation of the PEG loss' and the GDU models' influence on CGM forecasting resulted in a comprehensive codebase. It was developed to preprocess raw CGM data, train and evaluate the models described, as well as analyse the results.

Our implementation[2] was developed in Python 3.9 [52] and is modularized into three key components.

Preprocessing This module loads the raw CGM dataset, executing the artifact handling methodologies elucidated in Section 2.1.2. Post-cleaning, the data is split—either into 5 folds at random or based on the intrinsic treatment classifications of the individuals. This processed data is subsequently saved, ready for the forecasting module's consumption.

Forecasting Centralized within this module are the model architectures, metric definitions, loss function formulations, and training protocols. Given the appropriate model architecture, hyperparameters, and data partitioning scheme, the system trains the models, subsequently computing the evaluation measures. Both the refined models and their evaluative performance metrics are stored for subsequent analysis.

Analysis This module undertakes the systematic aggregation and comparison of performance metrics across diverse model trainings. It's equipped to compute and represent distances amid feature vectors and bases of GDU models. Moreover, it facilitates the generation of visual interpretations, showcasing exemplary forecasting outcomes, performance differences, and t-distributed stochastic neighbor embedding (t-SNE) embeddings.

Several libraries played crucial roles in the implementation of our methodologies. Pandas [53] was essential for CGM data preprocessing, while NumPy [54] handled array computations and metric calculations. Gaussian interpolation was handled by GPytorch [55] while scikit-learn

[2] https://github.com/andras-s/CGM-Forecasting-Robustness-and-Safety

[56] contributed to data preprocessing and t-SNE-based visualization of the GDU feature vectors and bases. Deep learning architectures. metrics, and loss functions were realized with PyTorch [57], complemented by CUDA's [58] GPU acceleration on NVIDIA's RTX 3090. Finally, visual findings were illustrated using Matplotlib [59] and Seaborn [60].

3. Results

3.1. Clinical relevance of the Parkes error grid loss

To determine if the extra efforts dedicated to implementing the PEG loss genuinely yield models with less clinically severe predictions, we trained the models detailed in Section 2.2.2 using both the NLL loss and the PEG loss. By employing a 5-fold cross-validation strategy, the LSTM and TM were initialized, trained, and evaluated on each fold. Details regarding the model and training hyperparameters are available in 2.2.2 and 2.3.2, respectively. As a point of reference, the performance of the t_0 model is also considered in the forthcoming results.

Given a model f, a sample (x, y) with $y = (y_1, \ldots, y_h)$, and the model's predictions $\mu = (\mu_1, \ldots, \mu_h)$ and $\sigma = (\sigma_1, \ldots, \sigma_h)$ a prediction (μ_i, σ_i) is classified as belonging to an area $X \in A, B, C, D, E$ of the PEG if the mean prediction μ_i falls within area X, i.e., $(y_i, \mu_i) \in X$. On each iteration of the 5-fold cross-validation the predictions on the reserved test data were categorized into the PEG areas A, B, C, D, E and the corresponding percentages of predictions per area were recorded. The mean and standard deviation of these percentages, calculated across the five folds, are reported in Table 3.1. We also provide the relative changes in percentages when transitioning from NLL loss to PEG loss, underscoring the clinical performance disparity between the two loss functions. Notably, no predictions fell into area E, so this area has been excluded from Table 3.1. To demonstrate the models' technical performance, we computed the mean and standard deviation of the trained models' RMSE and NLL using the same procedure described above for calculating percentages of predictions in the PEG areas. These results are presented in Table 3.2 and Table 3.3.

We summarize the results in Table 3.1, which shows the PEG percentage results of the models trained with and without the PEG loss. The introduction of the PEG loss significantly improved the mean percentage of predictions landing in the clinically severe area D, with a concurrent decrease in standard deviation. Almost universally, the use of the PEG loss also enhanced the mean percentage of predictions in area C. However, in the clinically benign areas, A and B, the mean percentages experienced only negligible changes.

Table 3.2 shows, that in terms of mean RMSE, the LSTMs outperformed the TMs, with both surpassing the t_0 baseline. The adoption of the PEG loss induced only marginal alterations to the mean RMSE. On shorter PHs of 30 and 60 minutes, models trained with the PEG loss generally displayed a minor improvement, while a slight increase in mean RMSE was observed for the 120-minute PH. The standard deviation improved for the TMs, although it increased for the LSTMs.

The mean NLL results are presented in Table 3.3. In the context of mean NLL, the LSTMs

Table 3.1.: 5-fold cross validation PEG percentage results. The mean test percentage per PEG area (standard deviation) and the relative change of the mean are reported. Best results according to the mean percentage are highlighted in **bold**.

		A	B	C	D
		30-min PH			
t_0		*96.104 (.225)*	*3.836 (.215)*	*0.059 (.020)*	*0.000 (.000)*
TM	NLL	97.222 (.215)	2.726 (.216)	0.053 (.018)	0.000 (.000)
	NLL+PEG	97.276 (.118)	2.675 (.105)	**0.048 (.015)**	0.000 (.000)
	rel change	*0.1*	*-1.8*	*-8.0*	*0.0*
LSTM	NLL	97.432 (.099)	2.501 (.100)	0.066 (.018)	0.000 (.001)
	NLL+PEG	**97.437 (.111)**	**2.494 (.113)**	0.068 (.011)	0.000 (.000)
	rel change	*0.0*	*-0.3*	*3.7*	*0.0*
		60-min PH			
t_0		*89.070 (.400)*	*10.408 (.326)*	*0.511 (.093)*	*0.011 (.006)*
TM	NLL	90.812 (.347)	8.645 (.333)	0.537 (.060)	0.006 (.003)
	NLL+PEG	91.003 (.309)	8.493 (.263)	**0.498 (.080)**	0.006 (.002)
	rel change	*0.2*	*-1.8*	*-7.2*	*-7.9*
LSTM	NLL	91.118 (.352)	**8.327 (.318)**	0.548 (.059)	0.007 (.004)
	NLL+PEG	**91.121 (.344)**	8.369 (.300)	0.507 (.089)	**0.003 (.002)**
	rel change	*0.0*	*0.5*	*-7.6*	*-59.1*
		120-min PH			
t_0		*77.633 (.672)*	*20.155 (.404)*	*2.052 (.268)*	*0.160 (.014)*
TM	NLL	79.671 (.518)	18.019 (.344)	2.171 (.212)	0.139 (.031)
	NLL+PEG	79.692 (.362)	18.169 (.213)	2.025 (.181)	0.114 (.016)
	rel change	*0.0*	*0.8*	*-6.7*	*-18.1*
LSTM	NLL	80.084 (.556)	**17.622 (.353)**	2.157 (.209)	0.137 (.018)
	NLL+PEG	**80.177 (.693)**	17.739 (.509)	**1.982 (.204)**	**0.102 (.016)**
	rel change	*0.1*	*0.7*	*-8.1*	*-25.6*

Table 3.2.: 5-fold cross validation RMSE results. The mean (standard deviation) test RMSE is reported. Best results according to the mean RMSE are highlighted in **bold**.

		30-min PH	60-min PH	120-min PH
t_0		*0.724 (.014)*	*1.216 (.034)*	*1.922 (.059)*
TM	NLL	0.634 (.019)	1.091 (.026)	1.700 (.034)
	NLL+PEG	0.623 (.014)	1.077 (.023)	1.708 (.031)
LSTM	NLL	0.609 (.006)	**1.062 (.022)**	**1.680 (.040)**
	NLL+PEG	**0.604 (.007)**	**1.062 (.023)**	1.685 (.043)

again outshone the TMs, but the t_0 model demonstrated a strong baseline, besting all other models on shorter PHs of 30 and 60 minutes. While the application of the PEG loss improved the mean NLL of the TMs on shorter PHs, the mean NLL marginally deteriorated for the TM on the longer PH and for the LSTM across all PHs. The NLL standard deviation

for the TMs improved with the use of the PEG loss, while there were no consistent changes for the LSTMs.

Table 3.3.: 5-fold cross validation NLL results. The mean (standard deviation) test NLL is reported. Best results according to the mean NLL are highlighted in **bold**.

		30-min PH	**60-min PH**	**120-min PH**
t_0		***-0.803 (.010)***	***0.089 (.034)***	*1.081 (.078)*
TM	NLL	-0.255 (.137)	0.287 (.062)	0.816 (.026)
	NLL+PEG	-0.324 (.117)	0.221 (.038)	0.828 (.020)
LSTM	NLL	-0.490 (.026)	0.133 (.042)	**0.752 (.037)**
	NLL+PEG	-0.466 (.031)	0.153 (.032)	0.765 (.040)

3.2. Domain generalization capabilities

To assess and enhance the LSTM model's robustness against changes in the population's diabetes treatment, we trained and evaluated the model and its adaptations as outlined in 2.4.1. Henceforth we refer to the LSTM model as the ERM single model to distinguish it from the LSTM ensemble model and the four versions of the GDU model (FT or E2E with CS or MMD). The preprocessed CGM data was split by the individuals' treatment types (BI, MDI, CSII, AP) into 4 distinct folds. Each fold, in turn, was set aside as test data, with the remaining folds used for training. This allowed us to evaluate how well the trained models would adapt to an unseen domain. Each training and evaluation procedure was performed five times with different random seeds to initialize the models. The mean and standard deviation of the NLL and RMSE are reported in Tables 3.4 and A.3, respectively. Model and training hyperparameters are provided in 2.4.1 and 2.4.2.

The results in Table 3.4 reveal that all models perform better in the DG setting when the unseen domain comprises BI or MDI and worse when the unseen domain is CSII or AP. Ranking the models' performance on different hold outs from best to worst yields: BI, MDI, AP, and CSII. The larger ERM ensemble model outperforms the ERM single model in all but one cases. In the exceptional case, a substantial surge in the standard deviation of the ERM single model's results suggests that the superior result might be attributed to an outlier in model performance. The GDU models trained via fine-tuning consistently outperformed the ERM ensemble model on the 30- and 60-minute prediction tasks, except for one instance. Regardless of the similarity measure used, the fine-tuned models surpassed the ERM ensemble models, with the most exceptional results achieved by models using negative MMD. For the 120-minute PH, the fine-tuned models excelled over the ERM single model but did not consistently surpass the ERM ensemble model. The difference in mean NLL between the fine-tuned GDU models and the ERM ensemble model on the 120-minute prediction task was minimal. The GDU models trained end-to-end (E2E) lagged behind all other models.

Table 3.4.: Cross domain NLL results. The mean (standard deviation) test NLL is reported. Best results according to the mean NLL are highlighted in **bold**.

		BI	MDI	CSII	AP
		30-min PH			
ERM	*Single*	*-0.6726 (.0507)*	*-0.5260 (.0718)*	*-0.3204 (.0315)*	*-0.3816 (.0620)*
	Ensemble	*-0.6756 (.0154)*	*-0.5640 (.0316)*	*-0.3642 (.0212)*	*-0.3600 (.0168)*
FT	CS	-0.7008 (.0230)	-0.5882 (.0382)	-0.3696 (.0208)	-0.3782 (.0241)
	MMD	**-0.7138 (.0211)**	**-0.5994 (.0402)**	**-0.3742 (.0299)**	**-0.3888 (.0259)**
E2E	CS	-0.5912 (.0247)	-0.4698 (.0453)	-0.2676 (.0159)	-0.2852 (.0090)
	MMD	-0.5918 (.0198)	-0.4594 (.0242)	-0.2714 (.0096)	-0.2790 (.0210)
		60-min PH			
ERM	*Single*	*-0.0286 (.0171)*	*0.0974 (.0233)*	*0.2760 (.0056)*	*0.2466 (.0080)*
	Ensemble	*-0.0618 (.0106)*	*0.0554 (.0121)*	*0.2456 (.0038)*	*0.2346 (.0097)*
FT	CS	-0.0692 (.0112)	0.0534 (.0133)	0.2396 (.0045)	0.2314 (.0085)
	MMD	**-0.0760 (.0130)**	**0.0504 (.0136)**	**0.2366 (.0052)**	**0.2250 (.0086)**
E2E	CS	0.0052 (.0072)	0.1234 (.0067)	0.2894 (.0059)	0.2808 (.0035)
	MMD	0.0110 (.0078)	0.1118 (.0188)	0.3004 (.0174)	0.2866 (.0120)
		120-min PH			
ERM	*Single*	*0.6260 (.0088)*	*0.7302 (.0097)*	*0.8844 (.0080)*	*0.7938 (.0077)*
	Ensemble	***0.6044 (.0045)***	***0.7110 (.0082)***	*0.8734 (.0106)*	***0.7816 (.0059)***
FT	CS	0.6050 (.0054)	0.7120 (.0099)	**0.8728 (.0122)**	0.7820 (.0053)
	MMD	**0.6044 (.0059)**	0.7124 (.0101)	0.8782 (.0134)	0.7856 (.0055)
E2E	CS	0.6398 (.0069)	0.7326 (.0038)	0.8922 (.0026)	0.8222 (.0141)
	MMD	0.6352 (.0047)	0.7396 (.0071)	0.8928 (.0075)	0.8110 (.0054)

3.3. Ablation study

When employing the PEG or GDU loss, as delineated in 2.3.1 and 2.4.1, respectively, the weight of the loss terms play a pivotal role in shaping the performance metrics of the resultant models. In this ablation study, we elucidate the interplay between the weights of loss terms and the performance metrics of the models exemplary for a 60-minute PH. We begin by examining the implications of varying the PEG loss weight (λ_{PEG}) on different model metrics before delving into the impact of the GDU loss weight parameters for domain similarity (λ_{OLS}) and sparse coding (λ_{L_1}).

Effect of the PEG loss weight. The magnitude of the PEG loss weight λ_{PEG}, determines the contribution of the PEG term in the total loss, given by

$$\mathcal{L}(y,(\mu,\sigma)) = \mathcal{L}_y(\mu,\sigma^2) + \lambda_{PEG}\hat{\mathcal{L}}_{PEG}(y,\mu). \tag{3.1}$$

We contrast models having $\lambda_{PEG} \in \{2^{-6}, 2^{-5}, \ldots, 2^{6}\}$[1] with the model devoid of the PEG loss, i.e., $\lambda_{PEG} = 0$, and portray how the test metrics' means – NLL, RMSE, and percentages per PEG area – alter. The bar plots in Figure 3.1 illustrate the mean relative changes of these metrics.

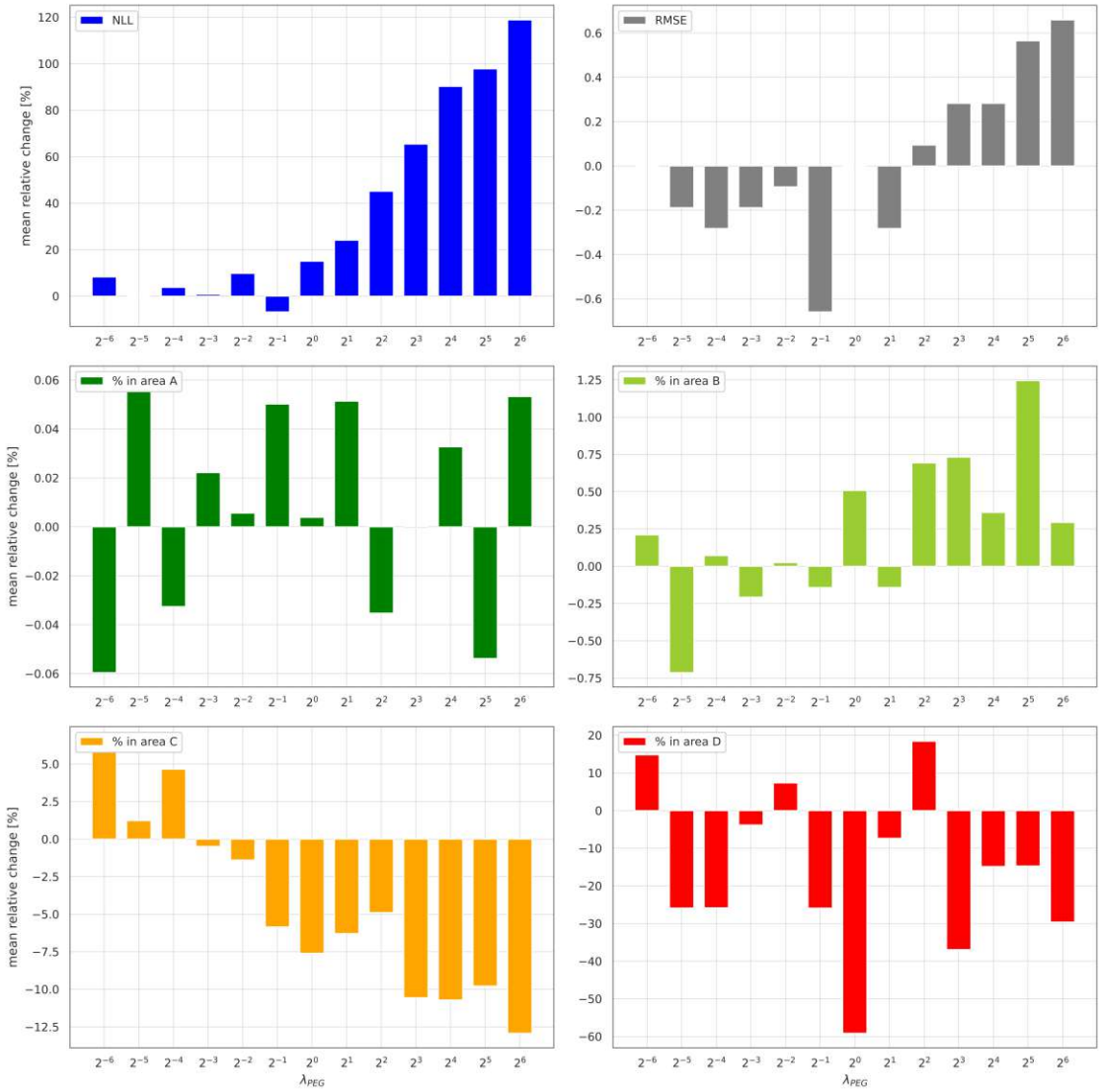


Figure 3.1.: Ablation study results depicting the impact of fluctuating PEG weight (λ_{PEG}) values from $2^{-6}, 2^{-5}, \ldots, 2^{6}$ on model performance. The bar plots represent mean relative changes in the NLL, RMSE, and percentage distribution across PEG areas (A, B, C, and D) compared to the respective metrics of a reference model without the PEG loss component ($\lambda_{PEG} = 0$). The influence of the PEG loss on model performance is illuminated, underlining the trade-offs between different metrics.

Our findings show that the mean NLL generally escalates when the PEG loss is employed, with this increase becoming more pronounced as the weight, λ_{PEG}, grows. This NLL rise is

[1] Starting with the initial value of $\lambda_{PEG} = 1$ the range was symmetrically increased until trends were not only visible, but confirmed with further range increases.

anticipated since a larger PEG weight in the total loss Equation 3.1 prioritizes minimizing $\hat{\mathcal{L}}_{PEG}(y, \mu)$ over the NLL term $\mathcal{L}_y(\mu, \sigma^2)$ during loss minimization. The relative change of mean NLL remains below 20% for weights ≤ 1 and exceeds 20% for weights > 1, reaching its peak of nearly 120% for a weight of $2^6 = 64$.

Similarly, the RMSE generally inflates with the weight, albeit at a lesser scale with a maximum increase less than 1%. Interestingly, for weights ≤ 1, the mean RMSE has improved compared to models not employing the PEG loss. The most significant reduction of -0.6% is achieved for $\lambda_{PEG} = 2^{-1}$, which appears to be an outlier, since neighbouring RMSE changes are of much smaller scale.

The mean relative change in the percentage of predictions in area A remains under 0.1%[2] for all weights, with no discernible pattern evident. The mean percentage of predictions in area B subtly hints at an increase with growing weight. For weights > 2, the mean percentage in area B surpasses that of the baseline models, with the maximum absolute change reaching 1.25% at $\lambda_{PEG} = 2^5$.

The mean percentage of predictions in area C is distinctly on a decline with the enlargement of the weight λ_{PEG}, hitting a maximum decrease of over 12.5% at the largest weight 2^6. The bar plot indicates that weights $\leq 2^{-4}$ lead to an uptick in the metric, whereas larger weights induce a decrease.

While the employment of the PEG loss generally results in a decrease in the mean percentage of predictions in area D, no consistent correlation with the weight magnitude is observable. The most substantial reduction of 59.1% is recorded at $\lambda_{PEG} = 1$.

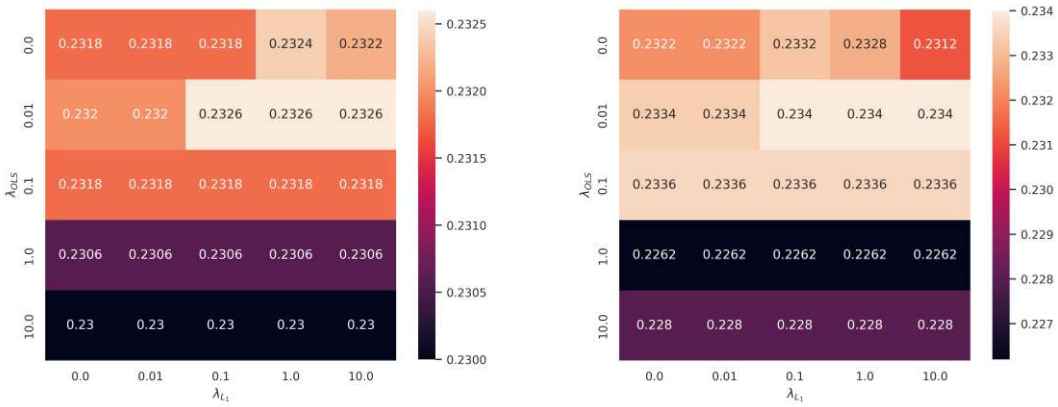
Effect of the GDU loss weights When training a GDU model g, the loss reads as

$$\mathcal{L}(y, (\mu, \sigma), g) = \mathcal{L}_y(\mu, \sigma^2) + \lambda_{OLS}\Omega_D^{OLS}(\|g\|_{\mathcal{H}}) + \lambda_{L_1}\Omega_D^{L_1}(\|\gamma\|), \quad \lambda_{OLS}, \lambda_{L_1} \geq 0.$$

Here, the term $\Omega_D^{OLS}(\|g\|_{\mathcal{H}})$ quantifies the distances between the elementary domain bases V_j and the elementary domains $\mathbb{P}_j$ in the RKHS. Therefore, the weight λ_{OLS} modulates the importance of this approximation during the model training process. The weight λ_{L_1} dictates the sparsity of the weight vectors β, resulting in fewer GDUs being active during prediction. Following Föll et al. [33], we manipulated the weight parameters $\lambda_{OLS}, \lambda_{L_1} \in 0, 0.01, 0.1, 1, 10$, tested the models using the AP data, and recorded the resultant mean test NLLs. The findings are visually represented as heatmaps in Figure 3.2.

Figure 3.2 shows that for both similarity metrics—CS and negative MMD—the resulting NLLs are robust across the entire parameter range, with only subtle differences noted. In both instances, the parameter λ_{OLS} exerts a greater influence than λ_{L_1}, suggesting that the approximation of the elementary domain bases is more critical than the sparse coding. As λ_{OLS} increases, the relevance of sparse coding diminishes. The optimal results are obtained for higher values of λ_{OLS}, specifically, $\lambda_{OLS} = 10$ for CS and $\lambda_{OLS} = 1$ for MMD.

[2]The scales of the relative changes for all metrics is differing, hence the relative changes in area A seem large visually while numerically they are insignificant.



(a) Mean NLL of GDU models utilizing CS

(b) Mean NLL of GDU models utilizing MMD

Figure 3.2.: The visualization of the test NLL of the GDU models with varying GDU loss weights λ_{OLS} and λ_{L_1}. The models were tested on AP data as the holdout with 5 randomized restarts. The reported NLL values are the averages over the 5 restarts. A 60-min PH was selected to represent the average case. Stable results across the whole parameter range are present. The significance of λ_{OLS} before λ_{L_1} as well as the increase in performance with the size of λ_{OLS} can be read from the visualizations.

4. Discussion

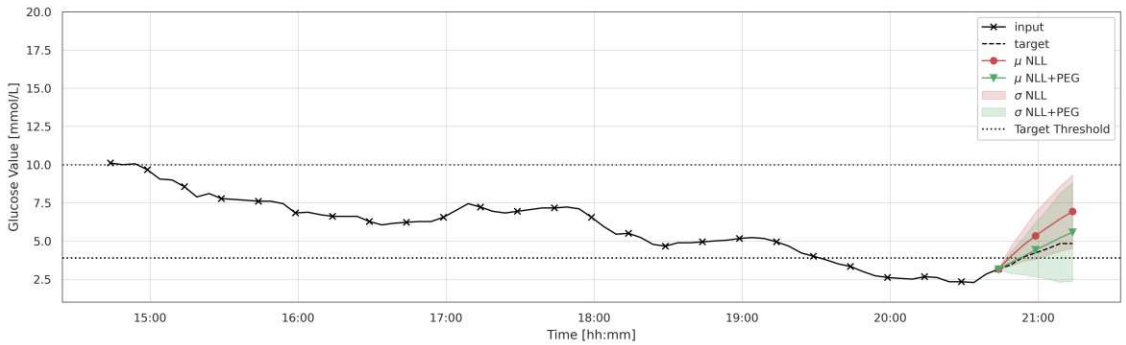
4.1. Interpretation of results

4.1.1. Evaluation of the PEG loss

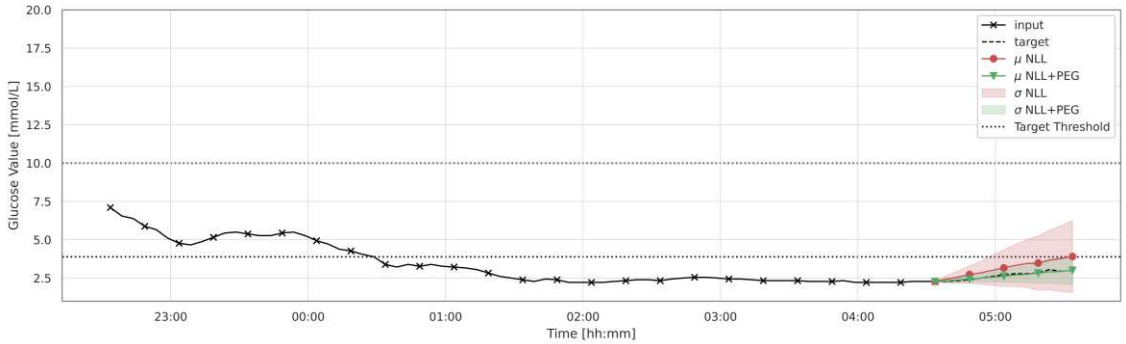
The comparative experiment on CGM forecasting performance, with and without the application of the PEG loss, demonstrated that models trained incorporating the PEG loss consistently produce fewer predictions that fall into the clinically severe areas C and D. The impact of this loss function is particularly evident in the longer PHs of 60 and 120 minutes, with a modest improvement observed for the shorter PH of 30 minutes. Further investigation of model performance within clinically severe PEG regions led to the sample forecasts depicted in Figure 4.1. Beyond merely numerical evidence of augmented prediction capacity in high-risk regions, this figure vividly exhibits a behavioral shift in the forecasts generated by the models. It demonstrates the prediction made by models trained with and without the PEG loss when facing scenarios with a potential for clinically severe predictions, such as low CGM levels at the last input times. In these situations, the models trained exclusively with the NLL tend to predict a convergence to a mean CGM level. However, models trained using the PEG loss yield more accurate predictions under the same conditions. Although these are merely representative forecasts, this behavioral shift was consistently observed during a comparative visual analysis of the forecasts, further highlighting the improved performance of the models in potential clinically severe scenarios.

The nature of the PEG loss suggests that the resulting models improve in terms of clinically severe predictions at the cost of overall technical performance. The PEG loss is appended to the NLL, enabling the model during training to minimize the overall loss by partially disregarding the NLL. As hypothesized, models trained with the PEG loss displayed enhanced performance in terms of clinical severity, with negligible variations in mean NLL, and even an observed improvement in mean RMSE.

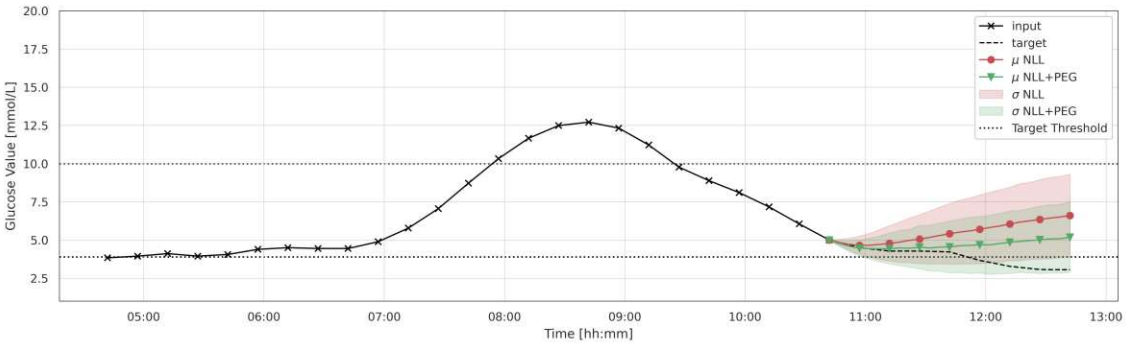
The ablation study in Section 3.3 analyzes the impact of the magnitude of the PEG loss weight λ_{PEG} on the resulting LSTM model performance metrics for a 60-minute PH. For the majority of λ_{PEG} magnitudes, the application of the PEG loss leads to a decrease in the mean percentage of predictions in area D, as these predictions result in large loss values coupled with significant loss gradients due to the design of the PEG loss. The low percentage of samples falling into area D demonstrates the high variability characteristic of this region. A clear trend is discernible in the more common and clinically relevant area C, where a consistent and significant reduction in prediction errors can be attained as the loss weight increases. While the expected increase in mean NLL can be confined to less than 20% for weights equal to or less than 1, a minor improvement in RMSE is observed



(a) An improved 30-minute forecast through the utilization of the PEG loss.



(b) An improved 60-minute forecast through the utilization of the PEG loss.



(c) An improved 120-minute forecast through the utilization of the PEG loss

Figure 4.1.: Visual representation of three sample forecasts from the LSTM model, with and without the PEG loss on the test set. In conditions that present a potential for severe predictions, the PEG loss trained models tend to predict more accurately. In contrast, the base models predict a convergence of the CGM values back to a mean level.

for weights equal to or less than 2. Consequently, the PEG loss weight range from 2^{-3} to 2 is considered optimal as it yields a desirable reduction in clinically severe predictions while maintaining a satisfactory technical performance level. The parameter λ_{PEG} offers a trade-off between the volume of clinically severe predictions and a potential decline in overall technical performance.

The added experimental computational complexity resulting from calculating the PEG loss instead of the NLL is detailed in Table 4.1. The runtime of each training epoch and the required number of epochs until model convergence were measured during the training of the models responsible for the results in Section 3.1. The models were trained on a NVIDIA Gigabyte GeForce RTX 3090. Despite a minor increase in the epoch runtime when implementing the PEG loss, it remains well within one standard deviation for both the transformer and the LSTM model. The number of epochs until convergence shows no consistent trend, indicating that the additional computational complexity associated with the PEG loss is negligible.

Table 4.1.: 5-fold cross validation runtime results. The mean (standard deviation) training time in seconds and the number of epochs until convergence are reported. Best results according to the mean are highlighted in **bold**.

		runtime per epoch [s]	**number of epochs**	**total runtime [s]**
TM	NLL	**1.25 (.12)**	1188 (522)	**1481 (639)**
	NLL+PEG	1.26 (.14)	1504 (126)	1910 (153)
LSTM	NLL	1.49 (.15)	1169 (127)	1743 (201)
	NLL+PEG	1.57 (.15)	**1103 (248)**	1738 (373)

4.1.2. Evaluation of the GDU results

Section 3.2 demonstrated the OOD performance disparities of various LSTM model modifications across different domains. It was observed that the prediction of CGM levels in individuals receiving modern treatments such as CSII and AP was more challenging when these treatment groups were not included during model training. Conversely, prediction was more accurate for older treatments like BI and MDI, when utilized as the hold out test data.

Comparing the daily CGM curves in Figure 2.1 and the domain characteristics in Table 2.1, it is clear that the BI domain, despite representing only 13% of the dataset, exhibits similarities with the MDI domain, which accounts for 44% of the data. This similarity likely contributes to the more accurate predictions in the BI domain, even when it was not directly trained upon. Conversely, the AP domain, comprising 11% of all data, displayed the largest disparity in terms of its characteristics and average daily CGM curve compared to the other domains. Nevertheless, model performances were better in AP than in CSII, likely due to the reduced glucose variability in the AP domain, making predictions generally more straightforward as major glucose level fluctuations are less common.

Ensemble methods, as stated in Wang et al. [61], can enhance the OOD performance of models, which is reflected in the superior performance of the ERM ensemble model compared

to the single ERM model. However, this improvement comes with the trade-off of increased complexity and longer runtimes, as detailed in Table 4.2. Against these ERM baselines, the GDU model was tested with both fine-tuning and E2E training. The fine-tuned model consistently outperformed the ERM ensemble model in 30- and 60-minute prediction tasks, suggesting that the performance boost is not solely due to the added complexity of the GDU model. In contrast, E2E trained models failed to deliver competitive predictions in terms of mean NLL, indicating that further refinement of model training hyperparameters may be necessary.

The robustness of the GDU model's performance to the specific choice of the parameters in the GDU loss λ_{OLS} and λ_{L_1}, as demonstrated in the ablation study, suggests its suitability for DG in CGM forecasting without the need for additional hyperparameter tuning. The study indicated that the magnitude of λ_{L_1}, therefore, sparse coding, wasn't particularly critical. This suggests that a subset of the available GDUs may be sufficient for accurate predictions. A comprehensive analysis of the learned elementary domains can be found in Section 4.1.3. The importance of λ_{OLS} in the accurate approximation of elementary domains in the RKHS is emblematic of its relevance to overall OOD performance.

The added DG performance of the methods outlined so far, comes at the cost of additional computational complexity. Table 4.2 collects the average runtimes during model training of all models on the-60 minute prediction task using CSII as a hold out with 5 restarts. Since for the fine-tuning of models pre-trained FEs are necessary, the reported metrics for FT do not directly compete with the other models' metrics. The ERM single model has the lowest computational cost and has the fastest computation time per epoch. The fine-tuning of GDU models is slower, but comparable to the ERM ensemble model, while the E2E GDU training has the longest average training time per epoch. Due to the constant improvement of the model to the overall best performance, the FT model takes the most epochs to converge.

The enhanced DG performance of the methods discussed comes at the expense of increased computational complexity, as highlighted in Table 4.2. Here, we presented the average training runtimes for all models on the 60-minute prediction task using CSII as a holdout, with 5 restarts. As the fine-tuning process necessitates pre-trained FEs, the metrics for FT do not directly compete with those of the other models. The ERM single model has the lowest computational cost and has the fastest computation time per epoch. The fine-tuning of GDU models is slower, but comparable to the ERM ensemble model, while the E2E GDU training has the longest average training time per epoch. As the FT model consistently improves to achieve the best overall performance, it requires the most epochs to converge.

4.1.3. Interpretability of elementary domains

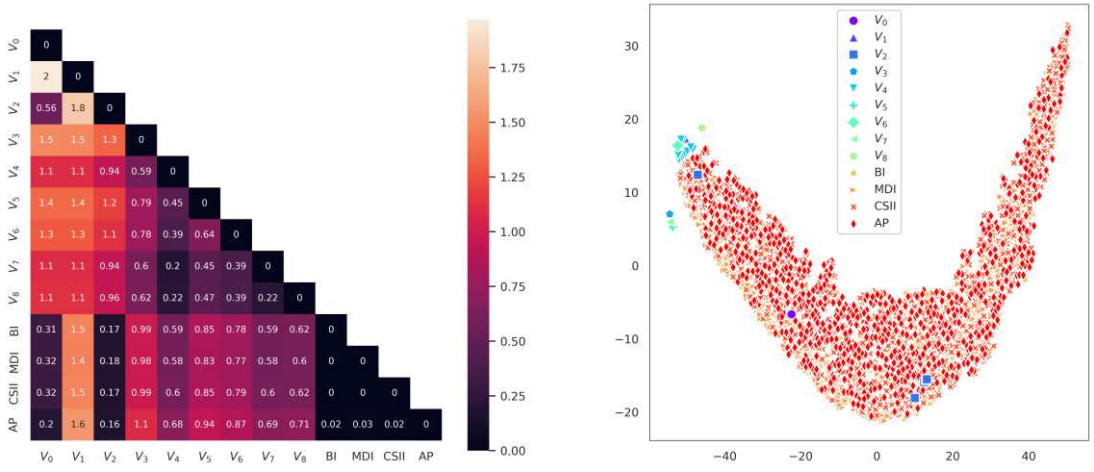
The learned parameters $V_1, \ldots, V_M$ of the GDU model can be interpreted as the learned representations of the elementary domains $\mathbb{P}_1, \ldots, \mathbb{P}_M$ of the feature vector space in the RKHS. To understand what elementary domains the GDU model is approximating and how these relate to the feature vectors we visualize the MMD between each domain basis V_j and domains in the feature vector space as well as a representative t-SNE of the bases and

Table 4.2.: 5-fold domain generalization runtime results on the CSII data averaged over 5 restarts with different random seeds. The mean (standard deviation) training time in seconds and the number of epochs until convergence are reported. Best results according to the mean are highlighted in **bold**.

		runtime per epoch [s]	number of epochs	total runtime [s]
ERM	*Single*	**1.48 (.16)**	846 (176)	**1253 (240)**
	Ensemble	1.83 (.19)	928 (195)	1697 (402)
FT	CS	2.06 (.18)	3382 (1054)	6959 (2144)
	MMD	2.05 (.18)	5925 (939)	12166 (1954)
E2E	CS	3.03 (.14)	**719 (105)**	2177 (340)
	MMD	3.10 (.17)	748 (108)	2315 (377)

feature vectors. A more detailed description of t-SNE, the parameters we used are provided in A.5. We visualize the feature vectors and bases of the GDU model that was fine-tuned to predict on a 60-minute PH with CSII as the hold out domain utilizing negative MMD as a similarity measure. The feature vectors of the training, validation and test data were calculated and visualized together.

Figure 4.2's heatmap (left) suggests that the feature vectors, produced by the feature extractor, are not significantly divergent across treatment domains. Figure 4.2 (right) shows that except for the bases V_0 and V_2 the bases lie outside of the feature vector cluster, indicating that the bases V_0 and V_2 approximate distributions within the feature vector space that the feature vectors from their neighbourhoods stem from. The bases V_1, V_3, V_5 and V_6 lie furthest away from the feature vector cluster, suggesting that they do not represent a certain population within the feature vectors, but rather an elementary domain shared by the feature vectors.

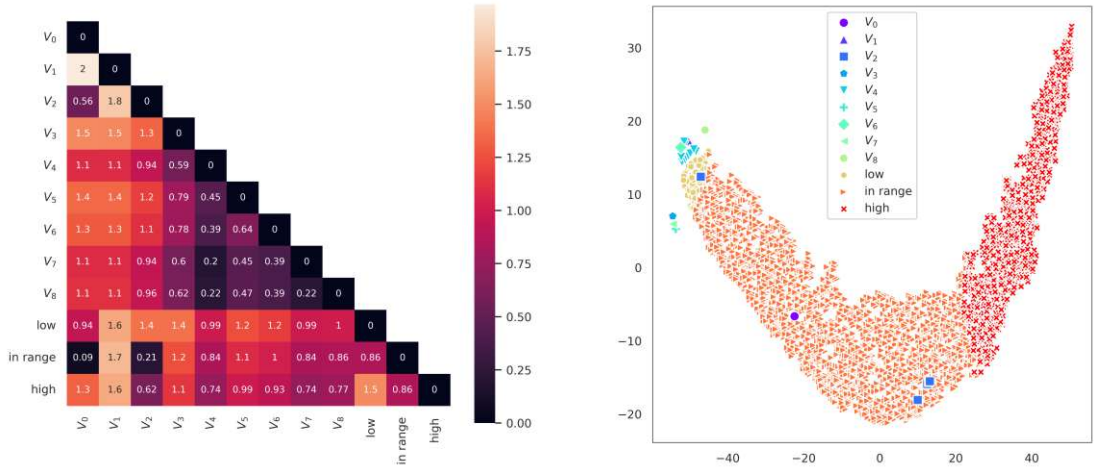


(a) MMD between learned bases and feature vectors

(b) t-SNE embedding of learned bases and feature vectors

Figure 4.2.: The visualizations of the similarity between the learned approximations of the elementary domain bases and the feature vectors of the GDU model categorized by treatment type. The FE does not differentiate between samples from the different treatment types. Bases outside the cluster approximate elementary distributions within the feature vector space, while bases enclosed by feature vectors represent the neighbouring samples.

Both Figures 4.3 (left) and (right) demonstrate that the feature extractor differentiates according to the last CGM measurement. The feature vectors are segregated into three categories based on the last CGM value in the input: *in range*, *low*, and *high*. The categories are determined by BG thresholds for being in range (3.9 to 10 mmol/l). The bases V_0 and V_2 are surrounded by feature vectors originating from inputs that were *in range*. These bases represent some subcategories within the *in range* category. The group of bases outside of the feature vector cluster is far from all categories as suggested by Figure 4.3's MMD heatmap (left), but closest to the *high* category. The t-SNE visualization in Figure 4.3 (right) accurately represents the property that these bases lie outside the feature vector cluster. The fact that the bases are far, in terms of MMD, from all categories suggests that the bases do not represent subcategories of feature vectors, but instead a more elementary property common to feature vectors across all categories. The visual proximity of these bases to the *low* category in the t-SNE plot might be due to a small subcategory within the *low* category that exceeds the proximity of all other subcategories. Given that very low CGM levels have a natural lower limit and high CGM values are unbounded and might fluctuate significantly, the model might reasonably focus on enhancing accuracy in a subcategory of very-low CGM levels which are easier to predict.



(a) MMD between learned bases and feature vectors

(b) t-SNE embedding of learned bases and feature vectors

Figure 4.3.: The visualizations of the similarity between the learned approximations of the elementary domain bases and the feature vectors of the GDU model categorized by the magnitude of a samples last CGM value. The FE does differentiate between samples with different CGM levels. Bases outside the cluster approximate elementary distributions that are present mainly in the *low* CGM category, while bases enclosed by feature vectors represent a subcategory of the *in range* category.

4.2. Limitations

Dataset. CGM data represents the glucose concentration in the interstitial fluid, inherently lagging behind the true BG concentration by an average of 4 minutes [4]. Additionally, CGM devices' readings are prone to errors, with the mean absolute relative difference to the blood glucose being up to 9.8% for modern devices [62]. Therefore, the optimal prediction of

CGM data is a slightly lagging and error-prone representation of the true BG trajectory. In applications such as an alert system or AP, the decisions made based on these predictions are continually influenced by these prediction errors, which could result in subtle alterations in the subsequent decisions [63].

The collected CGM data was collected under real-world conditions in open-label field studies, where individuals were monitoring and intervening based on their BG levels. Consequently, the models trained on this data are tailored to predict future CGM levels under the assumption that individuals are actively influencing their BG. However, when striving for fully autonomous BG management via an AP, CGM predictions would benefit from training on data that depicts glucose dynamics without human intervention.

In the preprocessing of the CGM data, we employed the algorithm from Baysal et al. [38] to filter out PISAs. The false-positive and true-positive rates for this process are 3.36% and 81.05%, respectively. This process results in data smoothing, potentially removing some rare, yet crucial scenarios. Models trained on such data may perform suboptimally when encountering situations similar to those discarded. Therefore, applications reliant on these models' predictions would need to identify and pause in these situations. For this reason, we chose a preprocessing method capable of detecting this rare scenarios in real-time, ultimately enhancing the overall practicality of our approach.

The split of the data into different domains according to the individuals' treatment types was suggested by Inselspital's medical staff and is in line with the recommendations from Finlayson et al. [11]. The analysis of model performance across these domains provides insight into the potential performance decrease of prediction models that comes with the introduction of new treatment types. The medical reasoning, the analysis of metrics as well as average daily CGM trajectories 2.1.1 indicate that these domains are in fact affected by domain shifts. Also the change in model performance when testing on an unseen treatment type suggests that the domains are inherently different, but a closer analysis of the magnitude of the shifts between the domains remains to be conducted.

Modeling. The LSTM model architecture was initially developed and tuned on the Ohio T1DM dataset [20, 43], which contains only individuals with type 1 diabetes, contrasting with our diverse dataset with various diabetes types. Further, the architecture was originally designed for individual-level training and testing, while we sought broader applicability, employing a population-wise approach. Although hyperparameter tuning might enhance the performance of the LSTM and TM for our dataset and tasks, we avoided it to prevent overfitting and to maintain the models' general applicability.

PEG evaluation. As a result of the open-label study design the performance evaluation using the PEG does not include predictions in area E, since standard CGM devices do not record measurements below 2.5 mmol/L, which is necessary for a prediction to fall into this area. Moreover, the high standard deviation of results in area D results from a scarcity of data in these clinically dangerous situations. Any production-level model should be thoroughly trained and tested in these scenarios.

GDU. In this work our aim was to show the general applicability of the methods to the use case of CGM forecasting and therefore potentially overfitting models to the given dataset was avoided at the cost of performance. Additional hyperparameter tuning may yield better performance for the GDU models [64], especially for the E2E model. The superiority of the fine-tuned models even in this case is strong evidence for the applicability of the proposed methods. The performance of the GDU models comes with added complexity as is showcased in 4.2.

The GDU models were trained initialized with 9 learning machines and therefore 9 domain bases as a result of finding the optimal number if clusters to cluster the FE output following the framework of Föll et al. [33]. The visualizations in 4.1.3 show, that some of the learned bases are very similar in terms of MMD. This indicates that unnecessary complexity is present with potentially marginal additions in performance. Remedying this are propositions for methods in 5 that result in only a necessary amount of bases that additionally are spread further apart from each other, representing more diverse elementary domains.

5. Conclusion and outlook

5.1. Conclusion

Within the expansive corpus of glucose prediction literature, our analysis discerns a critical need for predictions that transcend mere statistical relevance and bear direct clinical pertinence. Concurrently, an under-explored research gap has been discerned, rooted in the domain shift inherent to CGM data patterns, instigated by the advent of novel treatments.

Extensive research exists on glucose prediction, yet we identified the need for clinically more relevant predictions and the research gap characterized by the shifting domain of CGM data, due to the introduction of new treatment types. In response to these challenges, this research delineates two pivotal research questions.

RQ1 How can we integrate the clinical severance of erroneous predictions into a loss function for CGM forecasting deep learning models and to what extent does this loss reduce the amount of clinically severe errors?

In this thesis we derived 5 desirable properties of a loss function for CGM forecasting that accounts for the clinical severance of prediction errors in terms of the PEG. Based on the desired properties we developed the PEG loss with the aim to reduce the number of predictions that may result in clinically critical actions. Integrating this function into the training process of cutting-edge CGM forecasting models has proven successful in consistently reducing clinically severe errors. While the PEG loss does introduce an additional complexity and a small change in technical performance, these are negligible compared to its clinical benefits. The PEG loss also incorporates an interpretable parameter, offering a trade-off between technical and clinical performance, with our findings indicating a particular range for this parameter where both performances stay within acceptable bounds.

RQ2 To what extent can existing robust deep learning methods be utilized to make existing CGM forecasting models robust against distribution changes resulting from the population's diabetes treatment?

Our investigation further extends to the robustness of an LSTM model against distribution shifts caused by changes in diabetes treatment across the population. We found that the models exhibit limited generalization performance when trained without data from individuals undergoing specific treatment types. To enhance the models' domain generalization capabilities, we tested the performance of GDU models, trained in different modes, against a baseline ensemble model. The results demonstrated that fine-tuning GDU models with pre-trained LSTM feature extractors consistently improved robustness against distribution

shifts. Additionally, an ablation study of the GDU models' extra parameters suggested that the model's results are stable across a wide range of hyperparameters, thereby rendering hyperparameter tuning unnecessary.

While the GDU model does bring an increase in computational complexity, it compensates with greater interpretability. The learned bases of the GDU model represent subspaces of the feature vector space. Visual analysis of the similarity between the FE output and the GDU bases revealed that the FE consistently prioritizes an input's last glucose measurement and does not distinguish between treatment domains. The learned GDU bases either denote elementary domains in the feature vector space or highlight specific common subcategories of feature vectors.

In conclusion, the findings of this study emphasize the potential of the PEG loss and the GDU models to significantly improve the reliability and robustness of CGM forecasting systems. Future research should build upon these results to further refine these models, potentially leading to even better management of diabetes treatments.

5.2. Outlook

To enhance the applicability of the PEG loss and to further optimize the performance of CGM forecasting models in medically critical situations, we need more data in extreme scenarios such as glucose readings below 2.5 mmol/l. To create fully autonomous glucose management systems, these situations must be thoroughly explored to mitigate the risks of system failure during critical circumstances. Furthermore, data illustrating glucose dynamics without any individual or system intervention is required for the development of completely automated systems. However, the collection of data from potentially hazardous situations or instances without interference should be conducted under strict supervision from medical experts to safeguard the participating individuals.

The probabilistic forecasts produced by our models carry information regarding the models' certainty about a prediction, yet this (un)certainty is currently not incorporated into the PEG loss. To refine this, predictions with severe medical implications made with high model certainty should incur an increased loss, while those with significant uncertainty should have a reduced loss. One method to achieve this could involve calculating the probabilities for a prediction's distribution to generate a sample falling into the PEG areas and subsequently deriving a weighted sum of these probabilities.

The general applicability of the GDU models was shown by the superior performance of the fine-tuned models which in theory suggests that the poor performance of the E2E trained models could be improved using hyperparameter tuning, since the set of trained models through E2E training is a superset of the fine-tuned models. Hyperparameter optimization was omitted, but might be considered in future work.

The general applicability of the GDU models was demonstrated by the superior OOD performance of the fine-tuned models. This suggests that the underperformance of the E2E trained models might be improved through hyperparameter tuning, as the set of

models trained through E2E training encompasses the fine-tuned models. Although we omitted hyperparameter optimization in this study, it might be worth considering in future work.

The computed distances and visualizations 4.2 between the learned bases of the GDU model indicate that some bases are closely situated, hinting at potential redundancy in the resulting models. Conversely, some elementary domains may lack representation. To address this, future work could introduce a loss term that quantifies the proximity of bases to each other, i.e., during model training, one could aim to maximize the distance between bases up to a point. To further eliminate redundancies, one could start with a minimal set of GDUs and dynamically expand the model with additional GDUs during training until the variance between bases reaches a plateau.

Bibliography

[1] Gojka Roglic. „WHO Global report on diabetes: A summary“. In: *International Journal of Noncommunicable Diseases* 1.1 (2016), pp. 3–8.

[2] Hong Sun et al. „IDF Diabetes Atlas: Global, regional and country-level diabetes prevalence estimates for 2021 and projections for 2045“. In: *Diabetes research and clinical practice* 183 (2022), p. 109119.

[3] Edward W Gregg, Naveed Sattar, and Mohammed K Ali. „The changing face of diabetes complications“. In: *The lancet Diabetes & endocrinology* 4.6 (2016), pp. 537–547.

[4] Roy W Beck et al. „Advances in technology for management of type 1 diabetes“. In: *The Lancet* 394.10205 (2019), pp. 1265–1273.

[5] Rui Ma et al. „Recent advancements in noninvasive glucose monitoring and closed-loop management systems for diabetes“. In: *Journal of Materials Chemistry B* 10.29 (2022), pp. 5537–5555.

[6] Silvia Oviedo et al. „A review of personalized blood glucose prediction strategies for T1DM patients“. In: *International journal for numerical methods in biomedical engineering* 33.6 (2017), e2833.

[7] Ashenafi Zebene Woldaregay et al. „Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes“. In: *Artificial intelligence in medicine* 98 (2019), pp. 109–134.

[8] Vladimir Vapnik. *The nature of statistical learning theory*. Springer science & business media, 1999.

[9] Kaiyang Zhou et al. „Domain generalization: A survey“. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022).

[10] Andrew Wong et al. „External validation of a widely implemented proprietary sepsis prediction model in hospitalized patients“. In: *JAMA Internal Medicine* 181.8 (2021), pp. 1065–1070.

[11] Samuel G Finlayson et al. „The clinician and dataset shift in artificial intelligence“. In: *New England Journal of Medicine* 385.3 (2021), pp. 283–286.

[12] Clara Mosquera-Lopez et al. „Leveraging a big dataset to develop a recurrent neural network to predict adverse glycemic events in type 1 diabetes“. In: *IEEE journal of biomedical and health informatics* (2019).

[13] MZ Wadghiri et al. „Ensemble blood glucose prediction in diabetes mellitus: A review“. In: *Computers in Biology and Medicine* 147 (2022), p. 105674.

[14] Taiyu Zhu et al. „IoMT-enabled real-time blood glucose prediction with deep learning and edge computing“. In: *IEEE Internet of Things Journal* 10.5 (2022), pp. 3706–3719.

[15] Simone Del Favero, Andrea Facchinetti, and Claudio Cobelli. „A glucose-specific metric to assess predictors and identify models“. In: *IEEE transactions on biomedical engineering* 59.5 (2012), pp. 1281–1290.

[16] Xia Yu et al. „Deep transfer learning: a novel glucose prediction framework for new subjects with type 2 diabetes“. In: *Complex & Intelligent Systems* (2021), pp. 1–13.

[17] Shengwei Luo and Chunhui Zhao. „Transfer and incremental learning method for blood glucose prediction of new subjects with type 1 diabetes“. In: *2019 12th Asian Control Conference (ASCC)*. IEEE. 2019, pp. 73–78.

[18] Alessandro Aliberti et al. „A multi-patient data-driven approach to blood glucose prediction“. In: *IEEE Access* 7 (2019), pp. 69311–69325.

[19] Touria El Idriss et al. „Predicting blood glucose using an LSTM neural network“. In: *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE. 2019, pp. 35–41.

[20] John Martinsson et al. „Blood glucose prediction with variance estimation using recurrent neural networks“. In: *Journal of Healthcare Informatics Research* 4.1 (2020), pp. 1–18.

[21] Iván Contreras et al. „Using Grammatical Evolution to Generate Short-term Blood Glucose Prediction Models.“ In: *KHD@ IJCAI*. 2018, pp. 91–96.

[22] Josep Vehí et al. „Prediction and prevention of hypoglycaemic events in type-1 diabetic patients using machine learning“. In: *Health informatics journal* 26.1 (2020), pp. 703–718.

[23] Francesco Prendin et al. „Forecasting of glucose levels and hypoglycemic events: head-to-head comparison of linear and nonlinear data-driven algorithms based on continuous glucose monitoring data only“. In: *Sensors* 21.5 (2021), p. 1647.

[24] Giacomo Cappon et al. „Individualized Models for Glucose Prediction in Type 1 Diabetes: Comparing Black-box Approaches To a Physiological White-box One“. In: *IEEE Transactions on Biomedical Engineering* (2023).

[25] Chiara Zecchin et al. „How much is short-term glucose prediction in type 1 diabetes improved by adding insulin delivery and meal content information to CGM data? A proof-of-concept study“. In: *Journal of diabetes science and technology* 10.5 (2016), pp. 1149–1160.

[26] Ashish Vaswani et al. „Attention is all you need“. In: *Advances in neural information processing systems* 30 (2017).

[27] Qingsong Wen et al. „Transformers in time series: A survey“. In: *arXiv preprint arXiv:2202.07125* (2022).

[28] Haoyi Zhou et al. „Informer: Beyond efficient transformer for long sequence time-series forecasting“. In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 35. 12. 2021, pp. 11106–11115.

[29] Shiyang Li et al. „Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting“. In: *Advances in neural information processing systems* 32 (2019).

[30] William L Clarke et al. „Evaluating clinical accuracy of systems for self-monitoring of blood glucose“. In: *Diabetes care* 10.5 (1987), pp. 622–628.

[31] Andreas Pfützner et al. „Technical aspects of the Parkes error grid“. In: *Journal of Diabetes Science and Technology* 7.5 (2013), pp. 1275–1281.

[32] Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. CRC press, 2012.
[33] Simon Föll et al. „Gated Domain Units for Multi-source Domain Generalization“. In: *arXiv preprint arXiv:2206.12444* (2022).
[34] Tilmann Gneiting and Matthias Katzfuss. „Probabilistic forecasting“. In: *Annual Review of Statistics and Its Application* 1 (2014), pp. 125–151.
[35] Daniel Sarewitz, Roger A Pielke, and Radford Byerly. *Prediction: science, decision making, and the future of nature*. Island Press, 2000.
[36] Brett D Mensh et al. „Susceptibility of interstitial continuous glucose monitor performance to sleeping position“. In: *Journal of diabetes science and technology* 7.4 (2013), pp. 863–870.
[37] Nihat Baysal et al. „Detecting sensor and insulin infusion set anomalies in an artificial pancreas“. In: *2013 American Control Conference*. IEEE. 2013, pp. 2929–2933.
[38] Nihat Baysal et al. „A novel method to detect pressure-induced sensor attenuations (PISA) in an artificial pancreas“. In: *Journal of diabetes science and technology* 8.6 (2014), pp. 1091–1096.
[39] Zeinab Mahmoudi et al. „Fault and meal detection by redundant continuous glucose monitors and the unscented Kalman filter“. In: *Biomedical Signal Processing and Control* 38 (2017), pp. 86–99.
[40] Kezhi Li et al. „GluNet: A deep learning framework for accurate glucose forecasting“. In: *IEEE journal of biomedical and health informatics* 24.2 (2019), pp. 414–423.
[41] Rob J Hyndman and George Athanasopoulos. *Forecasting: principles and practice*. OTexts, 2018.
[42] Sepp Hochreiter and Jürgen Schmidhuber. „Long short-term memory“. In: *Neural computation* 9.8 (1997), pp. 1735–1780.
[43] Cindy Marling and Razvan Bunescu. „The OhioT1DM dataset for blood glucose level prediction: Update 2020“. In: *CEUR workshop proceedings*. Vol. 2675. NIH Public Access. 2020, p. 71.
[44] Jonas Gehring et al. „Convolutional sequence to sequence learning“. In: *International conference on machine learning*. PMLR. 2017, pp. 1243–1252.
[45] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. „Layer normalization“. In: *arXiv preprint arXiv:1607.06450* (2016).
[46] Kevin Plis et al. „A machine learning approach to predicting blood glucose levels for diabetes management“. In: *Workshops at the Twenty-Eighth AAAI conference on artificial intelligence*. 2014.
[47] Chiara Zecchin et al. „Jump neural network for real-time prediction of glucose concentration“. In: *Artificial Neural Networks*. Springer, 2015, pp. 245–259.
[48] Rishi J Desai et al. „Comparison of machine learning methods with traditional models for use of administrative claims with electronic medical records to predict heart failure outcomes“. In: *JAMA network open* 3.1 (2020), e1918962–e1918962.
[49] Qi Wang et al. „A comprehensive survey of loss functions in machine learning“. In: *Annals of Data Science* (2020), pp. 1–26.
[50] Masashi Sugiyama and Motoaki Kawanabe. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation*. MIT press, 2012.
[51] Krikamol Muandet et al. „Kernel mean embedding of distributions: A review and beyond“. In: *Foundations and Trends® in Machine Learning* 10.1-2 (2017), pp. 1–141.

[52] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[53] Wes McKinney et al. „Data structures for statistical computing in python“. In: *Proceedings of the 9th Python in Science Conference*. Vol. 445. Austin, TX. 2010, pp. 51–56.

[54] Charles R. Harris et al. „Array programming with NumPy“. In: *Nature* 585 (2020), pp. 357–362. DOI: 10.1038/s41586-020-2649-2.

[55] Jacob Gardner et al. „Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration“. In: *Advances in neural information processing systems* 31 (2018).

[56] Fabian Pedregosa et al. „Scikit-learn: Machine learning in Python“. In: *Journal of machine learning research* 12.Oct (2011), pp. 2825–2830.

[57] Adam Paszke et al. „PyTorch: An Imperative Style, High-Performance Deep Learning Library“. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[58] NVIDIA, Péter Vingelmann, and Frank H.P. Fitzek. *CUDA, release: 10.2.89*. 2020. URL: https://developer.nvidia.com/cuda-toolkit.

[59] John D Hunter. „Matplotlib: A 2D graphics environment“. In: *Computing in science & engineering* 9.3 (2007), pp. 90–95.

[60] Michael Waskom et al. *mwaskom/seaborn: v0.8.1 (September 2017)*. Version v0.8.1. Sept. 2017. DOI: 10.5281/zenodo.883859. URL: https://doi.org/10.5281/zenodo.883859.

[61] Jindong Wang et al. „Generalizing to unseen domains: A survey on domain generalization“. In: *IEEE Transactions on Knowledge and Data Engineering* (2022).

[62] Viral N Shah et al. „Performance of a factory-calibrated real-time continuous glucose monitoring system utilizing an automated sensor applicator“. In: *Diabetes technology & therapeutics* 20.6 (2018), pp. 428–433.

[63] Charlotte K Boughton and Roman Hovorka. „New closed-loop insulin systems“. In: *Diabetologia* 64 (2021), pp. 1007–1015.

[64] Rémi Bardenet et al. „Collaborative hyperparameter tuning“. In: *International conference on machine learning*. PMLR. 2013, pp. 199–207.

[65] Eric Schulz, Maarten Speekenbrink, and Andreas Krause. „A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions“. In: *Journal of Mathematical Psychology* 85 (2018), pp. 1–16.

[66] Laurens Van der Maaten and Geoffrey Hinton. „Visualizing data using t-SNE.“ In: *Journal of machine learning research* 9.11 (2008).

[67] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. „How to use t-SNE effectively“. In: *Distill* 1.10 (2016), e2.

A. Appendix

Contents

A.1. Participant characteristics

The dataset is large compared to the well-studied and publicly available dataset Ohio T1DM. Additionally, it represents a broader range of individuals with diabetes by including a range of different types of diabetes, treatments, and CGM devices as presented in Figure A.1. The dataset contains data from individuals with type 1, type 2 and other types (pancreatogenic, MODY, GDM, MIDD, IDM and posttransplant) of diabetes. The diabetes treatments were categorized by the Inselspital's medical staff as follows: BI = basal insulin only, MDI = multiple daily injections, CSII = continuous subcutaneous insulin infusion, AP = artificial pancreas (individuals equipped a MiniMed 670G, 770G, or 780G insulin pump).

A.2. PISA parameter analysis

The method for PISA detection presented in Baysal et al. [38] and used in this thesis introduces the three parameter sets: *trial*, *nominal*, and *cautious*. The parameter settings result in algorithms differing in their trade-off between true and false positive rates as shown in Table A.2. The algorithm with all three sets was applied to our dataset and the resulting data loss is also reported. We selected the *nominal* set to minimize data loss, whilst maximizing the true positive rate.

A.3. Interpolation with Gaussian process regression

Gaussian process regression is a Bayesian method for regression that can be used to approximate a wide range of functions [65]. Assume an unknown signal function $f : \mathbb{R}^n \to \mathbb{R}$

Table A.1.: Categorization of the 370 individuals in the dataset and size of each category in percent. (TXDM = Type X diabetes mellitus; BI = basal insulin only; MDI = multiple daily injections; CSII = continuous subcutaneous insulin infusion; AP = artificial pancreas)

	category	**%**
sex	male	65
	female	35
age	16 - 25	12
	25 - 50	50
	50 - 75	34
	75 - 87	3
diabetes type	T1DM	78
	T2DM	15
	other	7
treatment	BI	11
	MDI	31
	CSII	29
	AP	27
	other	2
HbA1c [%]	non-diabetic ($\leq$ 6.0)	8
	in control (6.0 - 7.0)	30
	monitor closely (7.0 - 8.5)	47
	elevated (8.5 - 10.5)	10
	seriously elevated ($\geq$ 10.5)	5
CGM sensor	Medtronic	49
	FreeStyle Libre	33
	Dexcom	18

Table A.2.: Parameter sets and the resulting true and false positive rates in Baysal et al. [38] as well as the data loss in our dataset. (True positive rate, $TPR = TP/P = TP/(TP + FN)$, and false positive rate, $FPR = FP/N = FP/(FP + TN)$)

parameter set	TPR **(%)**	FPR **(%)**	**data loss (%)**
trial	82.3	5.0	6.4
nominal	81.1	3.4	5.1
cautious	63.6	1.7	2.4

and some noise $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon^2)$, the objective is to estimate $y = f(x) + \epsilon$ at different inputs $x \in \mathbb{R}^n$. Previously collected observations $\{\mathbf{X}, \mathbf{y}\}$ with $\mathbf{X} = (x_1, \ldots, x_t)$ and $\mathbf{y} \in \mathbb{R}^t$ are used to update the belief about the unknown signal f. An estimation for $f(x)$ a point x, given previously observed data $\{\mathbf{X}, \mathbf{y}\}$ is given by the weighted sum

$$m_t(x) \sum_{i=1}^{t} w_i k(x_i, x)$$

with a chosen kernel function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ and weights collected in the vector $\mathbf{w} = \left(K(\mathbf{X}, \mathbf{X}) + \sigma_\epsilon^2 \mathbf{I}\right)^{-1} \mathbf{y}$, where

$$K(\mathbf{X}, \mathbf{X}) = \begin{bmatrix} k(x_1, x_1) & \ldots & k(x_1, x_t) \\ \vdots & \ddots & \vdots \\ k(x_t, x_1) & \ldots & k(x_t, x_t) \end{bmatrix}$$

and $\mathbf{I} = \text{diag}(1, \ldots, 1)$. For more detail, the interested reader is referred to [65]. Parameter optimization via some version of gradient descent is necessary to fit the values for σ_ϵ^2 and the parameters of the chosen kernel function to the given data.

We used the GPyTorch [55] implementation's Adam optimizer with an exponentially decaying learning rate with several kernels (i.e., RBF, Matérn and spectral mixture) to identify interpolations of our irregularly sampled CGM data. In all cases the quality of interpolation at a given point x was related to the amount of available data points surrounding x. So the CGM sequences were interpolated very well at the center of each sequence, but the behavior at the first and last sequence points was poorly captured due to the lack of surrounding CGM measurements. Considering that the last sequence points together with the slope at that point is crucial for short-term forecasting, this method resulted in interpolated data that later yielded forecasting models that were performing poorly compared to the models that were trained with the linearly interpolated data. Additionally, through the interpolation with the GP the interpolated time series was smoothed to some extent due to the assumed noise ϵ that comes with the method. Since we also had to tackle the irregular sampling times of the target CGM sequences, the interpolation method had to be applied to the target sequences as well, leading to smoothed targets. Smoother targets might lead to unrealistic forecasting results, because of the reduced variability, which in turn might suggest unrealistic results compared to real-world scenarios.

For these reasons and the added complexity reduction, we decided to only pursue linear interpolation for the re-sampling of the irregularly sampled CGM data.

A.4. Calculation of products in a RKHS

Given the RKHS $\mathcal{H}$ of real-valued functions on the feature space $\tilde{\mathcal{X}}$ with a reproducing kernel $k : \tilde{\mathcal{X}} \times \tilde{\mathcal{X}} \to \mathbb{R}$ and the vectors $u, v \in \tilde{\mathcal{X}}$ the scalar product between u and v in $\mathcal{H}$ can be calculated as

$$\langle \phi(u), \phi(v) \rangle_{\mathcal{H}} = k(u, v).$$

Furthermore, given $V = (v_1, \ldots, v_N)$ with $v_1, \ldots, v_N \in \tilde{\mathcal{X}}$, the product between u and V in $\mathcal{H}$ is

$$\langle \phi(u), \phi(V) \rangle_{\mathcal{H}} = \frac{1}{N} \sum_{i=1}^{N} k(u, v_i).$$

Lastly, given $W = (w_1, \ldots, w_N)$ with another set of vectors $w_1, \ldots, w_N \in \tilde{\mathcal{X}}$, the product between V and W in $\mathcal{H}$ reads as

$$\langle \phi(V), \phi(W) \rangle_{\mathcal{H}} = \frac{1}{N^2} \sum_{i,j=1}^{N} k(v_i, w_j).$$

A.5. t-SNE of feature vectors and GDU bases

In theory, as stated in Föll et al. [33] the learnable GDU bases $V_1, \ldots, V_M$ will approximate some underlying elementary distributions $\mathbb{P}_1, \ldots, \mathbb{P}_M$ of the feature vector space in the RKHS. *In practice*, we want to understand the structure of the feature vector space and how the learned bases V_i relate, i.e., how (dis-)similar they are, to certain regions of the feature vector space, to make the learned bases and therefore the GDU model more interpretable. Since the basis vectors and the feature vectors are 256-dimensional, standard visualization techniques to visualize the data's structure are not applicable. We resort to t-SNE to visualize the high-dimensional output of the FE together with the learned GDU bases $V_1, \ldots, V_M$. In the original publication [66] the ability of the method to reveal structure at many different scales and the importance thereof for high-dimensional data is outlined, which motivates the application of this method for our use case.

The set of vectors $V = \{w_1, \ldots, w_K\}$ we want to visualize together consists of the train and test set's feature vectors $\tilde{x}$ as well as the GDU basis vectors v_i^j. We aim to learn 2-dimensional representations W of the vectors in V that maintain the pairwise similarities between the original vectors and can be visualized in a 2-d plot. This is achieved by minimizing the cost defined as the Kullback-Leibler divergence between the distribution P of V and the distribution Q of W. For details the interested reader is referred to [66]. The similarities between any two vectors and therefore the cost is dependent on the perplexity parameter, which can be interpreted as a measure of the effective number of neighbours a vector has. Minimization is performed via gradient descent and therefore the learning rate and the number of iterations have to be set. We set the number of iterations to 5000, because in this use case convergence is always achieved after 5000 iterations. As suggested in Wattenberg, Viégas, and Johnson [67], we consider a range of perplexities and learning rates to find structure in our data. The perplexity and learning rate ranges considered are $\{25, 50, 100, 150, 250, 500, 1000\}$ and $\{50, 100, 150, 250, 500, 1000\}$, respectively.

We perform the analysis for the GDU model utilizing negative MMD as a similarity measure that was fine-tuned with the challenging CSII data as the hold-out test set. The resulting t-SNEs are visualized in Figure A.1 where the feature vectors were categorized by their treatment types. In most cases the embeddings form one large cluster with only some bases

outside of it. There is no visual indication that the distribution of feature vectors stemming from different treatment types are differing, from which we conclude that the FE is not able to capture the distribution shifts between the treatment types. This observation aligns with the results from Figure 4.2a. While the bases V_0 and V_2 are enclosed by neighbourhoods of the feature vectors, the remaining bases are located at the border or outside of one end of the feature vector cluster.

A more clear segmentation of the feature vectors becomes visible, when categorizing them by the input's last measurement as can be seen in Figure A.2. The bases enclosed in the feature vectors represent subcategories within the observations with last glucose measurements that are *in range*. The remaining bases lie outside the feature vector cluster. The bases outside of the cluster learn to approximate elementary distributions that are not representing a certain subcategory of the feature vectors, but an elementary attribute that is present in observations in all categories.

A.6. RMSE results of domain generalization experiment

For completeness we provide the supplementary RMSE results of the DG experiment detailed in 3.2. Similarly to the NLL results 3.4, the ERM single model shows decreasing performance on the newer treatment types CSII and AP. As expected, the ERM ensemble model outperforms the ERM single in almost all cases. Generally, the GDU models outperform the ERM ensemble model except for a small number of cases. In contrast to the NLL results the E2E models show competitive RMSE performance on all PHs, suggesting that the estimation of the mean CGM is of high quality, while the predicted variances are not accurate.

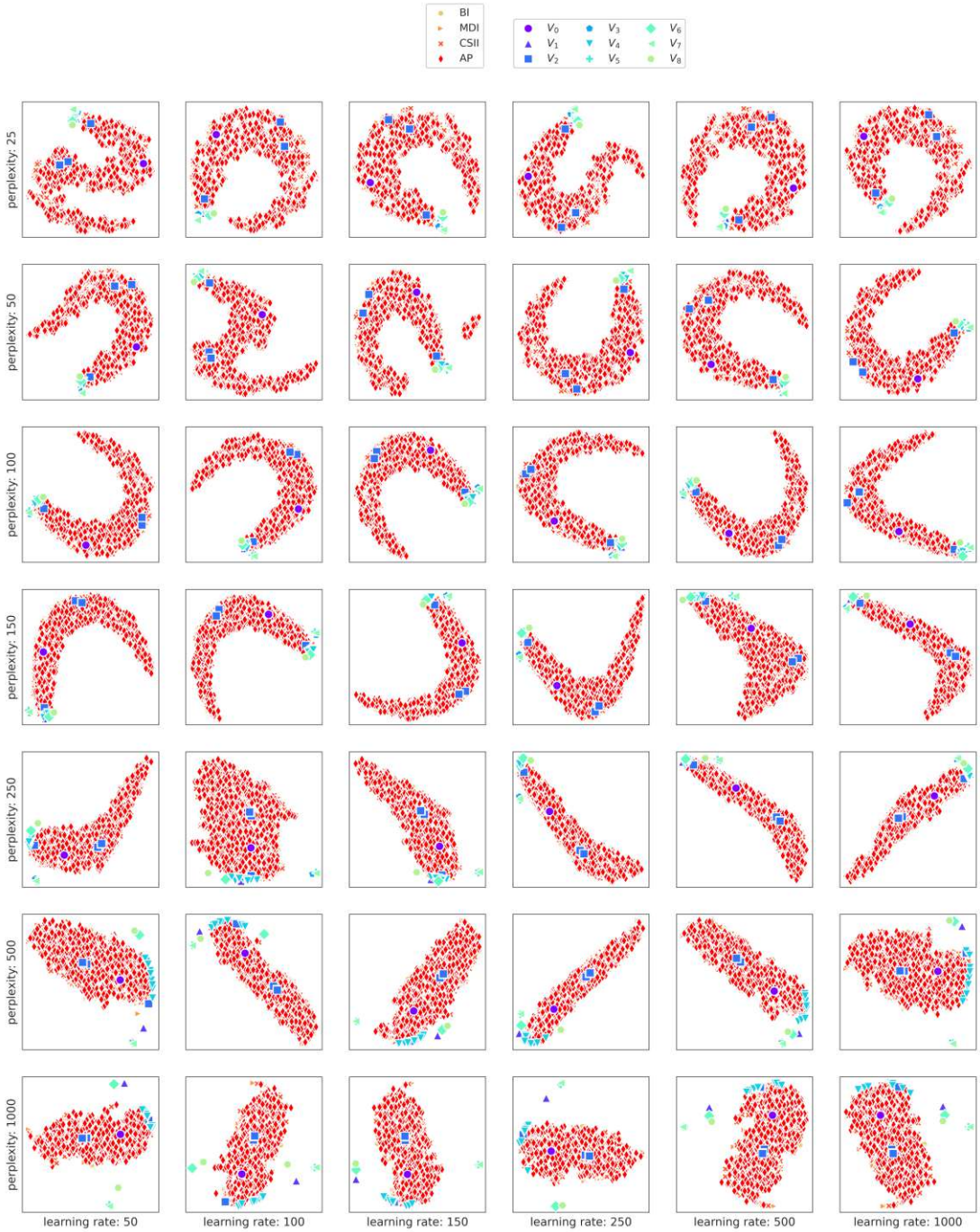


Figure A.1.: t-SNE visualizations of feature vectors and GDU bases at varying perplexity and learning rate parameters. The 2-dimensional representations illustrate the spatial relationships between feature vectors (categorized by treatment types) and the GDU bases $V_1, \ldots, V_M$. Despite differences in perplexity and learning rates, the visualizations consistently show one large cluster with a few outlying bases. Note the specific positioning of bases V_0 and V_2 within the feature vector neighborhoods, while the remaining bases tend towards the border or beyond the main cluster.

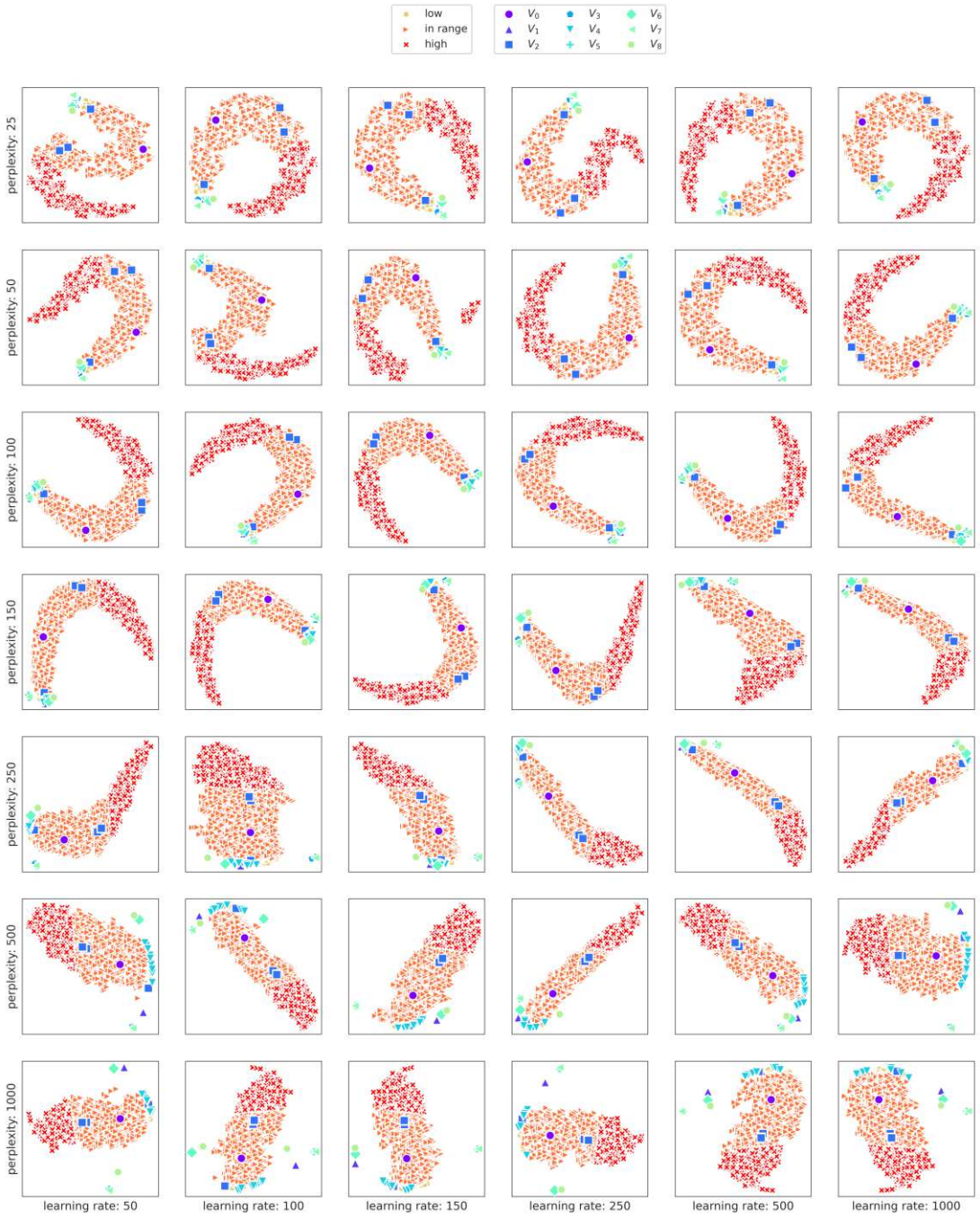


Figure A.2.: t-SNE visualizations of feature vectors and GDU bases categorized by the input's last glucose measurement. The 2-dimensional representations highlight a more distinct segmentation of the feature vectors. Bases enclosed within these clusters represent distinct subcategories within the observations. The bases outside the main cluster are presumed to approximate elementary distributions, representing an attribute that transcends individual categories and is present across all observations.

Table A.3.: Cross domain RMSE results. The mean (standard deviation) test RMSE is reported. Best results according to the mean RMSE are highlighted in **bold**.

		BI	MDI	CSII	AP
		30-min PH			
ERM	*Single*	*0.4970 (.0018)*	*0.5634 (.0034)*	*0.6962 (.0062)*	*0.6850 (.0039)*
	Ensemble	*0.4970 (.0017)*	*0.5628 (.0023)*	*0.6956 (.0065)*	*0.6840 (.0029)*
FT	CS	0.4968 (.0015)	0.5620 (.0024)	0.6966 (.0069)	0.6852 (.0032)
	MMD	0.4972 (.0015)	0.5622 (.0026)	0.6962 (.0067)	0.6848 (.0036)
E2E	CS	0.4960 (.0028)	**0.5592 (.0015)**	0.6992 (.0086)	0.6848 (.0016)
	MMD	**0.4940 (.0015)**	0.5600 (.0019)	**0.6874 (.0038)**	**0.6834 (.0024)**
		60-min PH			
ERM	*Single*	*0.9218 (.0031)*	*1.0164 (.0012)*	*1.1626 (.0064)*	*1.1424 (.0028)*
	Ensemble	*0.9214 (.0029)*	*1.0176 (.0012)*	*1.1580 (.0052)*	***1.1414 (.0030)***
FT	CS	0.9198 (.0032)	1.0158 (.0012)	1.1540 (.0046)	1.1448 (.0033)
	MMD	0.9196 (.0030)	1.0160 (.0011)	**1.1532 (.0048)**	1.1440 (.0028)
E2E	CS	**0.9160 (.0027)**	**1.0140 (.0018)**	1.1566 (.0050)	1.1482 (.0034)
	MMD	0.9180 (.0037)	1.0178 (.0024)	1.1592 (.0062)	1.1506 (.0042)
		120-min PH			
ERM	*Single*	***1.5320 (.0040)***	*1.6506 (.0035)*	*1.8064 (.0085)*	*1.7168 (.0116)*
	Ensemble	*1.5380 (.0032)*	*1.6490 (.0023)*	*1.8052 (.0058)*	*1.7198 (.0075)*
FT	CS	1.5360 (.0027)	1.6502 (.0017)	1.8034 (.0061)	1.7184 (.0088)
	MMD	1.5334 (.0031)	1.6502 (.0016)	1.8008 (.0050)	1.7170 (.0081)
E2E	CS	1.5366 (.0037)	**1.6456 (.0031)**	**1.7978 (.0050)**	1.7200 (.0102)
	MMD	1.5344 (.0045)	1.6480 (.0024)	1.8008 (.0073)	**1.7140 (.0076)**