

A Appendix

A.1 Examples Of Perturbation In Continuous Time

Figure 3 shows examples of the three perturbations applied to the same single-channel irregular time series. All perturbations are applied to the same area of that time series.

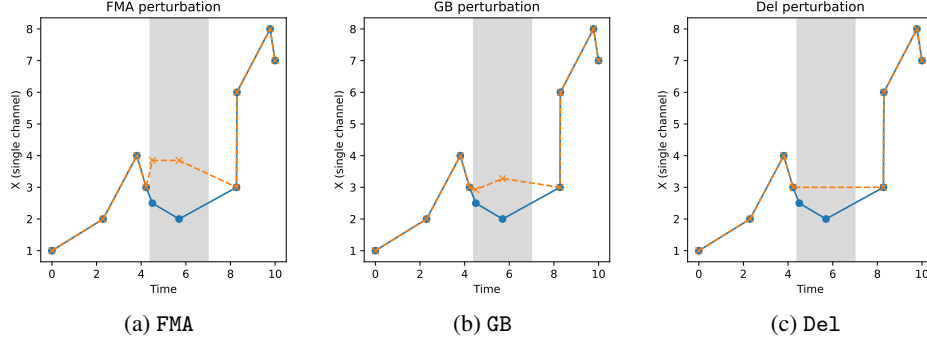


Figure 3: Examples of the three perturbations FMA, GB, and Del applied to same single channel irregular time series in the gray-shaded area. FMA insert the Moving Average, GB applies a Gaussian Blur, and Del deletes the points that lie inside the gray-shaded area.

A.2 Simulating The Deletion Of Data In A Differentiable Manner

Without knowing how exactly a model processes the data it receives as input, it is not possible to simulate that data had not been observed in a way that is differentiable with respect to the mask m , where m highlights which data points are being deleted. In such a case, m needs to be converted to a hard mask. Essentially, at some point during the process, m needs to function as an index. This means it has to be converted to an integer or boolean data type, which breaks the automatic differentiation graph in frameworks such as PyTorch.

For NCDE models, for instance, the way in which data is processed internally allows to (partially) simulate that data had not been observed. NCDE models model irregular data using differential equations. Via integration (i.e., the numerical ODE solver of one’s choice), the hidden states of the model are derived in continuous time. Given the nature of differential equations, the hidden states will therefore only update if the data changes. One can therefore delete the data observed at a certain timepoint by constructing a forward (or indeed backward) fill that overwrites said timepoint to duplicate one of the neighboring timepoints. Since this only involves multiplication and addition, we are still able to backpropagate w.r.t. m . However, note that this includes overwriting the time intensity channels, and also the entry in t . Thereby, one can alter that no new data had been observed at the chosen timepoint. This means simulating that no data had been observed at all at that timepoint—across all channels.

Going one step further, it is possible to simulate the deletion of data points in a way that is differentiable with respect to the deletion mask m for models such as `mtan` if one has access to model internals. Since `mtan` relies on attention modules to deal with missing data, it is possible to manipulate the attention mask to simulate that some further datapoints had not been observed. To do so, one requires access to the attention module. `mtan` uses scaled dot-product attention for some reference points (set somewhat arbitrarily to project the irregular data into a fixed dimension) and the timings of observed data points. The points one is trying to ‘delete’ can then be subtracted from the resulting attention mask or the corresponding part is simply multiplied by zero during the dot-product operation. These points then no longer influence the state of the model at the reference points, and thereby no longer have influence on the model output. Since this operation does not require a hard mask, and only involves multiplication and subtraction, it allows to simulate that data had not been observed if one is willing to access model internals.

Please note that we have not tested the ideas described above in practice. One would need to verify that they behave exactly as intended and indeed produce gradients useful for learning. While the

above strategies for mtan and NCDE do not break gradients, it is not guaranteed that they indeed feedback gradients that enable stable learning. If one is using an objective similar to Eq. 5, the gradients related to the penalty terms will not be zero, but gradients related to the error in f will likely oftentimes be zero which might still make learning difficult.

A.3 NeuroEvolution: performance, parameter counts, and runtime

In Table 6 we compare the runtime and performance when training MLP-F masks using different parameter counts. While performance varies based on parameter count, the time required for training remains relatively constant. All experiments were performed on a H200 GPU, where the used VRAM never exceeded 8 GB. In Table 6, the performance is averaged across the 10 runs, while the runtime is for all 10 experiments together. The results show that a much smaller mask with only 1906 parameters achieves similar performance to a mask close to 5000 parameters.

Training using NeuroEvolution takes 4–5 times longer than using gradient descent. As discussed in Section 5, we observe that masks parameterized as a Tensor of the same shape as X and d train in much fewer iterations using gradient descent, but require more iterations using NeuroEvolution. We’ve trained for 2,000 iterations using NeuroEvolution, and 32,000 iterations using gradient descent. Masks parameterized as MLP-F converge much better using NeuroEvolution but worse using gradient descent. We can only speculate about the reasons for this. Potentially, they are related to the assumptions algorithms such as PGPE make about the distributions of the parameters they are trying to optimize.

	Model Size			NeuroEvolution				Gradient Descent			
	Dim	L	Count	F1	Prec	Rec	Time	F1	Prec	Rec	Time
MLP-F	32	34	4914	0.622	1.000	0.452	2:18	0.250	0.796	0.167	0:27
MLP-F	32	32	4786	0.690	0.961	0.556	2:18	0.170	0.372	0.115	0:26
MLP-F	32	24	4274	0.638	0.774	0.557	2:08	0.170	0.569	0.112	0:26
MLP-F	16	24	1906	0.670	0.897	0.560	2:17	0.218	0.560	0.143	0:26
MLP-F	16	12	1522	0.584	0.835	0.490	2:18	0.157	0.377	0.108	0:27
MLP-F	8	24	914	0.539	0.717	0.515	2:17	0.107	0.381	0.068	0:26
MLP-F	8	12	722	0.497	0.806	0.429	2:13	0.106	0.353	0.070	0:27

Table 6: Model size, performance, and runtime averaged across 10 runs for the value-based Rare Feature problem when training MLP-F masks using the FMA perturbation across 10 runs. The performance is averaged across 10 runs, the runtime (Time) is given for all 10 runs in total in (h:mm), i.e., training an MLP-F mask with 4914 parameters 10 times takes 2 hours and 18 minutes using NeuroEvolution and only 27 minutes using gradient descent.

A.4 Regularizing Parameters

Table 7 and 8 display the chosen values for λ_1 and λ_2 for the Rare Feature and Rare Time problems for each of the two settings. We started with values for λ_1 that penalize the area covered by the mask too heavily and then decreased λ_1 until the mask returned reasonable results, where initially we restricted λ_2 such that $\lambda_1 = 10\lambda_2$ or $\lambda_1 = 100\lambda_2$. Once a value for λ_1 was set, we optimized further for λ_2 also allowing for smaller values of λ_2 .

Perturbation	Value-based		Temp-based	
	λ_1	λ_2	λ_1	λ_2
GB	0.01	0.001	0.1	0.000001
FMA	0.01	0.001	0.1	0.000001
Del	0.1	0.001	0.1	0.000001

Table 7: Values for λ_1 and λ_2 for the Rare Feature Problem reported in Table 2.

	Value-based		Temp-based	
Perturbation	λ_1	λ_2	λ_1	λ_2
GB	0.1	0.0001	0.00001	0.000001
FMA	0.1	0.0001	0.00001	0.000001
De1	0.1	0.0001	0.00001	0.000001

Table 8: Values for λ_1 and λ_2 for the Rare Time Problem reported in Table 3.

A.5 Updated Objective Function For Real-world Sepsis Task

For the sepsis prediction task, we adapted Equation 5 such that the loss term involving λ_1 changes from:

$$\lambda_1 \sum_C \int_0^T (1 - m(u)_c) du \quad (9)$$

to:

$$\lambda_1 \left(TS - \sum_C \int_0^T (m(u)_c) du \right), \quad (10)$$

where TS is the target size, i.e. in our case $100\% - 10\% = 90\%$.

For the sepsis task, the goal is to maximize the divergence in prediction while the mask should only alter a previously defined target size TS , for example, 10% of all observed time points. The goal for the previous tasks on synthetic data was to keep the prediction unchanged while altering as many data points as possible.

The full objective for the sepsis task then becomes:

$$\min_m -\mathcal{L}(f(t_n, x_n, d_n), (\tilde{t}_n, \tilde{x}_n, \tilde{d}_n)) + \lambda_1 \left(TS - \sum_C \int_0^T (m(u)_c) du \right) + \lambda_2 \sum_C \int_0^T |m(u)_c'| du. \quad (11)$$

A.6 Ablation Study for λ_1 and λ_2

We conducted an ablation study on the hyperparameters λ_1 and λ_2 for the De1, FMA, and GB attribution methods on the sepsis prediction task using mTAN. The results are reported for 10 cases predicted as sepsis-positive. Each cell contains two values: the first corresponds to the De1 odds change, and the second to the Val odds change. Note that these results can be higher than those reported for the full set of 100 sepsis-positive cases, as the top 10 predictions are typically easier to explain—more confident predictions tend to exhibit more pronounced feature relevance.

Table 9: Ablation on λ_1 and λ_2 for De1. Each cell shows De1 odds change | Val odds change.

λ_1	$\lambda_2 = 0$	$\lambda_2 = \lambda_1/100$	$\lambda_2 = \lambda_1/10$
0.01	6.24 7.06	10.34 7.94	5.95 8.27
0.1	9.13 8.44	9.91 7.81	6.51 6.82
0.5	10.75 8.83	9.05 6.89	8.09 7.42
1.0	12.92 8.42	13.64 7.19	10.99 8.48
10.0	11.24 6.81	8.50 7.55	8.69 7.30

A.7 Examples of Fitted Masks

To visualize how metrics such as F1-Score, Precision, and Recall represent the quality of fitted masks, Figure 4 shows examples of MLP-F masks fitted based on GB, FMA, and De1 perturbations using NeuroEvolution. The masks are fitted for the value-based Rare-Feature problem and correspond to what is displayed in Table 2 (100 time points and 50 features). Averaged across the 10 runs, the

Table 10: Ablation on λ_1 and λ_2 for FMA. Each cell shows Del odds change | Val odds change.

λ_1	$\lambda_2 = 0$	$\lambda_2 = \lambda_1/100$	$\lambda_2 = \lambda_1/10$
0.01	6.76 7.76	6.01 7.81	10.08 7.48
0.1	7.78 8.15	9.50 8.95	9.68 9.31
0.5	7.16 6.83	10.47 8.17	12.63 9.49
1.0	8.10 6.95	10.24 9.15	11.83 7.15
10.0	11.74 7.04	11.63 6.56	11.43 7.71

Table 11: Ablation on λ_1 and λ_2 for GB. Each cell shows Del odds change | Val odds change.

λ_1	$\lambda_2 = 0$	$\lambda_2 = \lambda_1/100$	$\lambda_2 = \lambda_1/10$
0.01	5.86 15.78	5.54 15.58	8.91 15.16
0.1	4.64 15.22	5.85 14.84	9.14 16.04
0.5	6.05 16.92	9.27 17.02	8.59 15.48
1.0	7.04 16.15	8.31 16.82	9.30 14.62
10.0	7.95 8.31	10.99 11.16	9.86 11.93

masks based on GB correspond to an F1 Score of 0.556, compared to 0.638 for masks based on FMA and 0.891 for masks based on Del. The ground truth saliency changes every run (i.e. changes row by row) but is the same for all perturbations (i.e., only one ground truth saliency per row).

A.8 Details On The Sepsis Prediction Task

We train a model according to [8] for sepsis prediction on real-world hospital data. Kidger et al. [8] feed cubic spline coefficients to the model. Since the computation of these coefficients takes long if there are missing data in X , they compute and save the coefficients once and simply load them during iterative training and testing. Since we alter the underlying data using our perturbations, we have to recompute the cubic spline coefficients at every iteration, which is very costly. Computing the coefficients according to Kidger et al. [8] takes approximately 30 seconds for a single triplet (t, X, d) . While this is largely parallelized, this significantly increases the time needed to explain a single prediction by the model. As per Table 6, 10 runs to using NeuroEvolution take around 2 hours and 18 minutes. This is roughly the time it takes to conduct one run for the real-world sepsis prediction task. Note that we have already decreased the iterations for PGPE from 2000 to 200. While the reduction to 200 iterations cuts computation time, it leads to poorer fitted saliency maps. We have not been able to quantify this drop in performance.

For other models such as mtan, where the raw data is fed into the models, perturbing data is much more straight forward and such experiments will be much closer to the Rare Feature and Rare Time examples in terms of required compute.

Table 12 shows the λ_1 and λ_2 values for the real-world sepsis prediction task using the NCDE model. Similar to before, we started with too large values of λ_1 and set $\lambda_1 = 10\lambda_2$ or $\lambda_1 = 100\lambda_2$. We then iteratively reduced λ_1 until we reached reasonable performance and then further finetuned λ_2 . Generally, we did not consider alternative values for λ_1 and λ_2 other than multiplies of 10^n for some positive or negative integer n .

For the mtan model, we simply set $\lambda_1 = 1$, and $\lambda_2 = 0$ (ablation study above).

	Pred: No Sepsis		Pred: Sepsis	
Perturbation	λ_1	λ_2	λ_1	λ_2
GB	1	0.01	1	0.01
FMA	0.00001	0.000001	0.00001	0.000001
Del	1	0.01	10	0.1

Table 12: Regularization parameters for the real-world sepsis prediction task using the NCDE model.

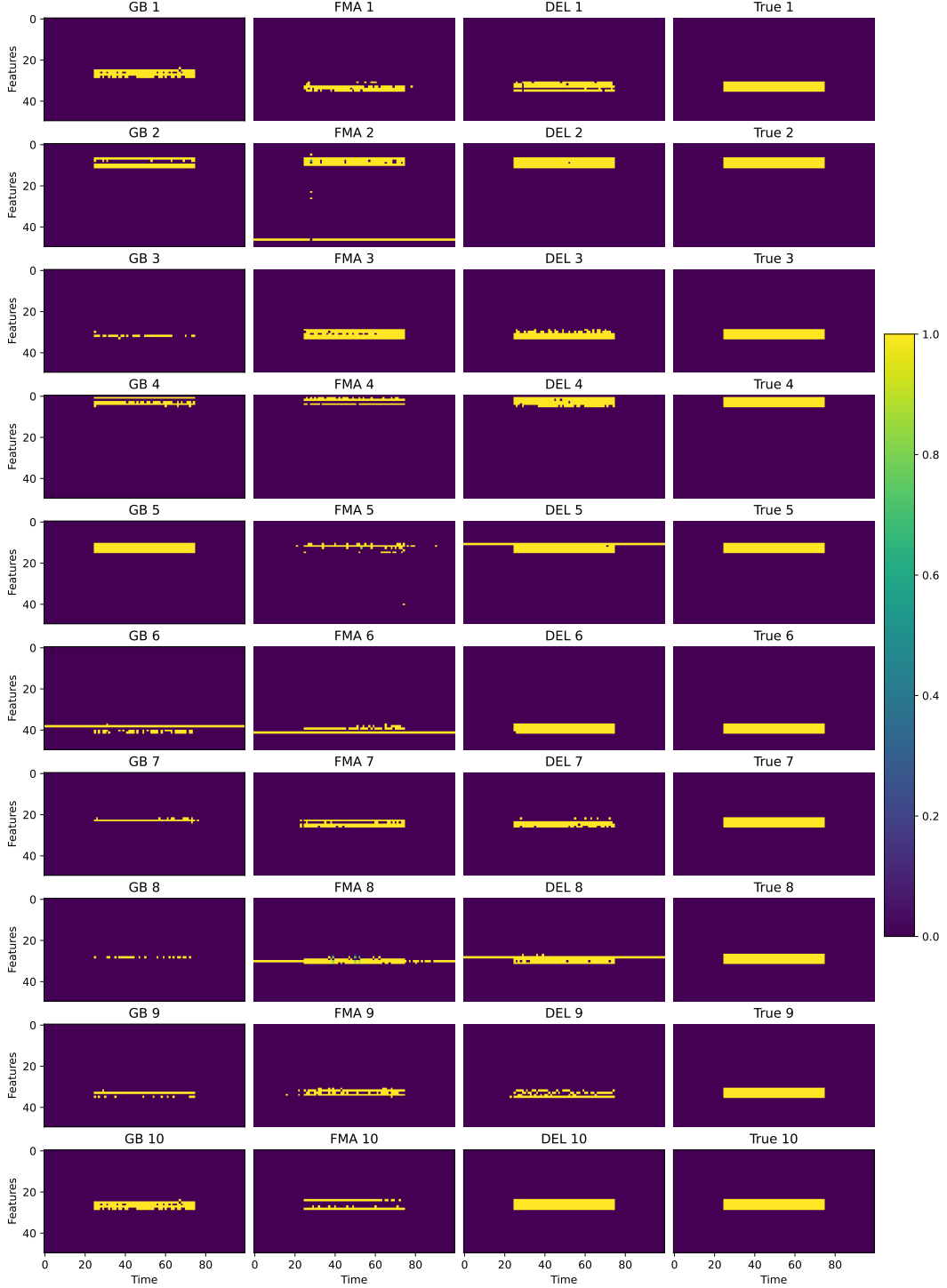


Figure 4: Examples of fitted masks using GB, FMA, and De1 for the value-based Rare Feature problem (MLP-F masks trained using NeuroEvolution). The masks correspond to the results is displayed in Table 2.

A.9 Mask Evaluation For Sepsis Prediction Task Using mtan Model

We evaluated for every feature of the data how often it was removed during the real-world sepsis task based on the mtan model. If a feature was removed very often, i.e., retention is low, it is estimated

to have a large effect on the prediction. In Table 13, we ordered all features by retention ratio, i.e., the most important features are listed first. It is a more extensive version of Table 5. In the Sepsis Imbalance column, we’ve calculated how much more often this feature was present for a patient who later becomes septic compared to a patient who does not become septic. High values indicate that the sheer presence of a particular feature might be a strong indicator of a patient later developing sepsis. This corresponds to decisions of the treating physicians who might decide to only collect certain information if a patient is at a high risk of developing sepsis.

Feature	Retention Ratio	Sepsis Imbalance	Description
Age	0.7912	1.0000	Patient age (y)
Height	0.7982	1.0000	Height (cm)
AST	0.8063	1.5256	Aspartate transaminase
ICUType	0.8196	1.0000	ICU unit type
ALP	0.8204	1.4433	Alkaline phosphatase
Cholesterol	0.8300	1.4276	Serum cholesterol
RespRate	0.8360	0.5092	Respiratory rate (bpm)
Creatinine	0.8422	1.0123	Serum creatinine
K	0.8589	1.0100	Potassium (mmol/L)
MechVent	0.8597	1.1880	Mechanical ventilation
Na	0.8610	1.0100	Sodium (mmol/L)
pH	0.8656	1.1483	Arterial pH
NI SysABP	0.8741	1.0183	Non-invasive systolic BP
Gender	0.8790	1.0000	Sex (F=0, M=1)
MAP	0.8793	1.0734	Mean arterial pressure
DiasABP	0.8846	1.0689	Diastolic BP
Weight	0.8852	1.0000	Weight (kg)
PaO2	0.8902	1.1642	Arterial oxygen (mmHg)
SysABP	0.8935	1.0689	Systolic BP
Temp	0.8946	1.0102	Temperature (°C)
Platelets	0.8956	1.0138	Platelet count
TroponinT	0.8998	1.4946	Cardiac troponin T
HR	0.9038	1.0102	Heart rate (bpm)
GCS	0.9057	1.0102	Glasgow Coma Scale
Urine	0.9058	0.9904	Urine output (mL)
Lactate	0.9093	1.3661	Blood lactate
ALT	0.9104	1.5256	Alanine transaminase
Glucose	0.9114	1.0191	Blood glucose
HCT	0.9174	1.0123	Hematocrit (%)
TroponinI	0.9186	2.6003	Cardiac troponin I
SaO2	0.9190	1.0734	O ₂ saturation (%)
NI DiasABP	0.9236	1.0200	Non-invasive diastolic BP
FiO2	0.9239	1.2318	Inspired O ₂ fraction (%)
NI MAP	0.9337	1.0200	Non-invasive MAP
HCO3	0.9370	1.0115	Bicarbonate (mmol/L)
BUN	0.9421	1.0123	Blood urea nitrogen
Bilirubin	0.9458	1.4518	Total bilirubin
PaCO2	0.9475	1.1642	Arterial CO ₂ (mmHg)
WBC	0.9551	1.0168	White blood cells
Albumin	0.9648	1.3987	Serum albumin
Mg	0.9721	1.0191	Magnesium (mmol/L)

Table 13: Features included in the real-world sepsis prediction task ordered by their estimation importance (most important features first) including the retention ratio, an indicator of structural bias (Imbalance) of this feature, and a short description.