LASSO regularisation within the LocalGLMnet architecture

By Ronald Richman and Mario V Wüthrich

Presented at the Actuarial Society of South Africa's 2022 Convention
Cape Town 25–28 October 2022

ABSTRACT
Deep learning models have been very successful in the application of machine learning methods, often out-performing classical statistical models such as linear regression models or generalised linear models. On the other hand, deep learning models are often criticised for not being explainable nor allowing for variable selection. There are two different ways of dealing with this problem, either we use post-hoc model interpretability methods or we design specific deep learning architectures that allow for an easier interpretation and explanation. This paper builds on our previous work on the LocalGLMnet architecture that gives an interpretable deep learning architecture. In the present paper, we show how group LASSO regularisation (and other regularisation schemes) can be implemented within the LocalGLMnet architecture so that we receive feature sparsity for variable selection. We benchmark our approach with the recently developed LassoNet of Lemhadri et al. (2021).

Keywords
Deep learning, neural networks, LocalGLMnet, regression model, variable selection, regularisation, LASSO, group LASSO, ridge regularisation, Tikhonov regularisation

CONTACT DETAILS
Ronald Richman, Old Mutual Insure and University of the Witwatersrand
Email: ronaldrichman@gmail.com
Mario V Wüthrich, RiskLab, Department of Mathematics, ETH Zurich
Email: mario.wuethrich@math.ethz.ch

1. INTRODUCTION

1.1 Background

Deep learning models have been used successfully in the application of machine learning to several domains with complex unstructured data, e.g., in computer vision, in natural language processing, and more recently, in generative modelling. These models achieve their success by applying the paradigm of representation learning to receive optimised features from raw data for a given prediction task. One disadvantage of the representation learning approach implemented through deep neural networks is that the resulting models are difficult to explain and interpret, since the predictions made by the networks result from a composition of several high-dimensional non-linear functions, and it is difficult to attribute model predictions to specific features within the raw data. Due to this, the extent to which feature selection can be performed by neural networks is rather limited compared with simpler regression models which operate directly on the raw data (without a representation learning step), e.g., in generalised linear models (GLMs) the Least Absolute Shrinkage and Selection Operator (LASSO) of Tibshirani (1996) can be applied for variable selection. This shortcoming of deep learning can be problematic for several reasons. First, predictions based on many features may be harder to explain than those based on fewer predictor variables, which causes difficulty when explanation of algorithmic decision-making is required by legislation or regulation, or if predictions are used in a scientific context, e.g., in medical decision-making. Second, removing irrelevant features from models often leads to greater predictive accuracy, as well as reduced training and inference times. Third, the collection and storage of many features can be prohibitively expensive, thus, practical considerations may favour simpler models that rely on only a few inputs.

In this manuscript, we study a feature selection approach in the recently proposed LocalGLMnet architecture of Richman and Wüthrich (2022). A LocalGLMnet uses a feed-forward neural (FFN) network to estimate the regression coefficients of a GLM for each instance individually. Since this model makes predictions using the additive structure of a GLM, the effect of each term can be related to the predictions, while still maintaining the advantages of representation learning. In this work, we study how feature selection approaches can be incorporated into the LocalGLMnet model, and compare and contrast the regularised LocalGLMnet model to other methods.

1.2 Related work

One approach to perform variable selection within machine learning and neural network models is to select variables based on the results of post-hoc model interpretability methods, which illustrate whether and how a feature contributes to the predictions of a model. Some examples of these are the Partial Dependence Plot (PDP) of Friedman (2001) or the Accumulated Local Effects (ALE) method of Apley and Zhu (2020). Other similar approaches within the neural network literature rely on the differentiability of deep learning models to estimate the gradients of these models which are then used in a variety of ways as explanations, see Merz et al. (2022) for a recent review of, and new contribution to, gradient-based methods.

A different direction in the literature is to design deep learning models with a structure that allows for interpretability, see the explainable neural network proposal of Vaughan et al. (2018) and the neural additive models of Agarwal et al. (2020). Both of these proposals rely on parallel neural networks fit on single variables at a time, to some extent limiting their applicability. Richman (2021) extends these proposals by combining a linear model with parallel networks fit on combinations of variables being identified using visual aids; see Richman (2021) for the Combined Actuarial eXplainable Neural Network (CAXNN) approach. The disadvantage of the CAXNN model is exactly this reliance on a manual specification of the parallel networks. If variables modelled using the parallel networks appear to make little contribution to the predictions of the model, these can be excluded, thus allowing for feature selection.

Relatively few methods have emerged within the neural network literature for directly performing variable selection. An important new proposal is the LassoNet of Lemhadri et al. (2021); this method is explained in more detail in the next section. A different, empirically-based test of variable significance is given in Richman and Wüthrich (2022), who exploit the additive structure of the LocalGLMnet architecture by adding a random feature to perform a sort of empirical Wald test for variable selection. Here, we focus on penalising the LocalGLMnet model loss function using well-known approaches for regularising linear regression models and GLMs: within the statistical learning literature, these typically are the ridge regression of Hoerl and Kennard (1970), also called Tikhonov (1943) regularisation, the LASSO[1] regularisation of Tibshirani (1996), and the combination of both of these in the ElasticNet proposal of Zou and Tibshirani (Zou & Hastie, 2005).

1.3 Review of LassoNet

We first review the LassoNet approach of Lemhadri et al. (2021), and then , in the following sections, we propose a new approach of regularisation and feature selection in neural

1 We call our proposal LASSO regularisation of the LocalGLMnet. Whereas the initial proposal of the LASSO was indeed for the linear regression model, this has been extended to GLMs, see Section 3.4 in Hastie et al. (2015).

network modelling. Denote by $\boldsymbol{x} \in \mathbb{R}^q$ the q-dimensional tabular features. The LassoNet combines a FFN network $\boldsymbol{x} \mapsto \Psi_W(\boldsymbol{x})$, having weights W, together with a skip connection $\boldsymbol{x} \mapsto \boldsymbol{\beta}^\top \boldsymbol{x}$, having regression parameter $\boldsymbol{\beta} \in \mathbb{R}^q$. This provides a regression function

$$\boldsymbol{x} \mapsto \boldsymbol{\beta}^\top \boldsymbol{x} + \Psi_W(\boldsymbol{x}). \tag{1.1}$$

The crucial part in this architecture is that one isolates a linear component of the model that is used as the basis for LASSO regularisation. For given data, Lemhadri et al. (2021) consider for parameter W and $\boldsymbol{\beta}$ an objective function $L(\boldsymbol{\beta}, W)$, and a ℓ_1-penalty term then implies feature sparsity

$$\underset{\boldsymbol{\beta}, W}{\text{minimise}}\, L(\boldsymbol{\beta}, W) + \eta \|\boldsymbol{\beta}\|_1, \tag{1.2}$$

$$\text{subject to} \|W_j^{(1)}\|_\infty \leq M|\beta_j| \text{ for all } j=1,\ldots,q, \tag{1.3}$$

for regularisation parameters $\eta > 0$ and $M > 0$. This latter parameter M is called the "budget" parameter, and where $W_j^{(1)}$ denotes the weights connecting the j-th input component x_j of feature $\boldsymbol{x}$ with the neurons in the first hidden layer of the FFN network architecture. The penalty term $\eta \|\boldsymbol{\beta}\|_1$ in (1.2) is the classical LASSO regularisation term that leads to sparsity in feature selection for large η, and the side constraint (1.3) gives a budget constraint that restricts the input signals of the corresponding feature components. Thus, if $\beta_j = 0$ is shrunk exactly to zero then the input signal of that component also is deactivated $W_j^{(1)} \equiv 0$.

1.4 Proposal of the regularised LocalGLMnet

The LocalGLMnet architecture introduced in Richman and Wüthrich (2022) looks similar to (1.1) in that it also contains a skip connection and a FFN network although it is fundamentally different. Namely, instead of adding a residual FFN network $\Psi_W(\boldsymbol{x})$ to the linear structure, we modify the linear part to a non-linear one by letting the regression parameter $\boldsymbol{\beta} = \boldsymbol{\beta}(\boldsymbol{x})$ be feature dependent, and a deep FFN network is used to model these feature dependent regression weights; this is formally introduced in Assumptions 2.1, below. Our proposal for regularisation will then be quite different from the LassoNet proposal, as our penalty term will not act on the model weights, but directly on the activations received from the network. This is described in detail in Section 3.1, and the different methods are illustrated in the diagrams in Section 3.4.

Organisation of this manuscript The rest of the manuscript is organised as follows. In the next section we introduce the LocalGLMnet architecture, and in the following Section 3, we discuss regularisation of this model to produce feature sparsity. The subsequent two sections, Sections 4 and 5, give synthetic and real data examples respectively. The final Section 6 concludes with a brief summary. The appendices contain code to implement the

regularised LocalGLMnet in the R language, descriptions of the data used in the examples and details on how the LassoNet was implemented as a comparison to the LocalGLMnet.

2. PROBLEM FORMULATION AND MODEL ARCHITECTURE

Assume that we have independent and identically distributed (i.i.d.) data $(Y_i, \boldsymbol{x}_i)$, $1 \le i \le n$, that has been generated from an unknown distributional model $(Y, \boldsymbol{x}) \sim F$, providing tabular features $\boldsymbol{x}_i \in \mathbb{R}^q$ and corresponding responses Y_i. The responses can either be class labels, (real-valued) random variables or random vectors. The general goal is to design a regression function $\boldsymbol{x} \mapsto \mu(\boldsymbol{x})$ that tries to predict Y as accurately as possible based on $\boldsymbol{x}$.

To this end we choose a suitable class $\mathcal{M}$ of potential regression functions $\boldsymbol{x} \mapsto \mu(\boldsymbol{x})$, and we seek to minimise the (empirical) prediction error

$$\arg\min_{\mu \in \mathcal{M}} \frac{1}{n} \sum_{i=1}^{n} L(Y_i, \mu(\boldsymbol{x}_i)), \tag{2.1}$$

for a given loss function L, also called scoring function. The prediction error in (2.1) is an empirical version of the (intractable) true prediction error $\mathbb{E}_F[L(Y, \mu(\boldsymbol{x}))]$, which is also called the expected generalisation loss. Gneiting and Raftery (2007) and Gneiting (2011) emphasise the importance of the specific choice of the loss function L, namely, that for an honest and truthful forecast evaluation one should choose a strictly consistent scoring function for L – refer to Definition 1 in Gneiting (2011).

There remains the choice of the class $\mathcal{M}$ giving us the potential regression functions over which we would like to optimise in (2.1). We choose for this class a family of LocalGLMnets of fixed complexity which is determined by the depth d of the FFN network, the widths of the hidden layers and the choice of the activation function in these hidden layers.

Assumptions 2.1 (LocalGLMnet) *Fix depth $d \ge 2$, numbers of neurons $q_1, \ldots, q_{d-1} \in \mathbb{N}$ and activation function $\phi : \mathbb{R} \to \mathbb{R}$ in the hidden layers, and set the output dimension equal to the input dimension $q_d = q$. This gives us a deep FFN network architecture*

$$\Psi_W : \mathbb{R}^q \to \mathbb{R}^q \qquad \boldsymbol{x} \mapsto \Psi_W(\boldsymbol{x}),$$

having network weights W. The LocalGLMnet regression function is defined by the generalised additive decomposition

$$\boldsymbol{x} \mapsto \mu(\boldsymbol{x}) = \mu_{W,\beta_0}(\boldsymbol{x}) = g^{-1}\Big(\beta_0 + \boldsymbol{\beta}(\boldsymbol{x})^\top \boldsymbol{x}\Big), \tag{2.2}$$

where $g : \mathbb{R} \to \mathbb{R}$ is a fixed strictly monotone and smooth link function, $\beta_0 \in \mathbb{R}$ is a bias, and where we set regression attentions $\boldsymbol{\beta}(\boldsymbol{x}) = \Psi_W(\boldsymbol{x})$.

As class $\mathcal{M}$ we then choose the set of all LocalGLMnet regression functions $\boldsymbol{x} \mapsto \mu_{W,\beta_0}(\boldsymbol{x})$ having a fixed FFN network architecture and a fixed link function. This class

is parametrised by the network weights and bias (W, β_0). Thus, we consider the specific minimisation problem for (2.1)

$$\arg\min_{W,\beta_0} \frac{1}{n}\sum_{i=1}^{n} L\Big(Y_i, \mu_{W,\beta_0}(\boldsymbol{x}_i)\Big). \tag{2.3}$$

We give some remarks and interpretation.

Remarks 2.2

— If we replace $\boldsymbol{\beta}(\boldsymbol{x}) \equiv \boldsymbol{\beta} \in \mathbb{R}^q$ by a constant parameter, then (2.2) describes a classical GLM regression function with link function g. A linear regression model is obtained by the identity link $g(x) = x$; this is also the link choice in (1.1).

— Instead of considering a residual FFN network as in (1.1) we use the network $\Psi_W(\boldsymbol{x})$ to model the regression parameters $\boldsymbol{\beta}(\boldsymbol{x})$, and we call them regression attentions because they give more or less attention to specific feature values $\boldsymbol{x}$ in (2.2), this is inspired by the attention layers used in transformer models, see, e.g., Vaswani et al. (2017).

— We describe the intuition behind (2.2). Consider an individual component $1 \le j \le q$

$$\boldsymbol{x} \mapsto \beta_j(\boldsymbol{x}) x_j. \tag{2.4}$$

(1) If $\beta_j(\boldsymbol{x}) \equiv \beta_j$ is not feature dependent, we receive a GLM term in $\beta_j x_j$.

(2) If $\beta_j(\boldsymbol{x}) \equiv 0$, term $\beta_j(\boldsymbol{x}) x_j$ is dropped.

(3) If $\beta_j(\boldsymbol{x}) = \beta_j(x_j)$, term $\beta_j(x_j) x_j$ does not interact with any other variables $x_{j'}$, $j' \neq j$.

(4) Interactions can be studied by considering the gradient of $\beta_j(\boldsymbol{x})$

$$\nabla \beta_j(\boldsymbol{x}) = \Big(\partial_{x_1}\beta_j(\boldsymbol{x}), \ldots, \partial_{x_q}\beta_j(\boldsymbol{x})\Big)^\top \in \mathbb{R}^q. \tag{2.5}$$

Interpretations (1)–(4) need some care because we do not have identifiability. For example, we could receive a term

$$\beta_j(\boldsymbol{x}) x_j = x_{j'}, \tag{2.6}$$

by learning a regression attention $\beta_j(\boldsymbol{x}) = x_{j'} / x_j$. For this reason we say, e.g. in item (2), that term $\beta_j(\boldsymbol{x}) x_j$ should be dropped but feature component x_j may still play an important role in some attention weights $\beta_{j'}(\boldsymbol{x})$, $j' \neq j$. For completely dropping a variable x_j, a second model fitting step is necessary to see whether the validation loss does not increase if x_j is completely dropped from the model.

Our numerical experiments on different configurations have not manifested any such issues (2.6) as Stochastic Gradient Descent (SGD) fitting seems rather predetermined by the LocalGLMnet functional form (2.2), and it just tries to fit the attention weights $\boldsymbol{\beta}(\boldsymbol{x})$ around the pre-specified linear terms. We conclude from items (1)–(4) that (2.2) gives us an interpretable regression function. Item (2) provides us with a way of selecting or dropping individual terms (2.4) depending on the significance of $\beta_j(\boldsymbol{x})$. This now exactly motivates our regularisation approach.

— In order to have comparability between continuous features components x_j we always assume that they are centred with unit variance. We discuss the corresponding treatment of categorical variables later.

3. REGULARISATION OF ATTENTION WEIGHTS

3.1 Formulation of optimisation problems

Based on the discussion in Remarks 2.2, it is rather straightforward to introduce regularisation to include or exclude individual terms in the additive decomposition (2.2). In what follows, we can consider the ridge regularisation of Tikhonov (1943), the LASSO regularisation of Tibshirani (1996) or the combined ElasticNet regularisation of Zou and Hastie (2005). The latter motivates the consideration

$$\underset{W,\beta_0}{\arg\min} \frac{1}{n}\sum_{i=1}^{n} L\Big(Y_i, \mu_{W,\beta_0}(\boldsymbol{x}_i)\Big) + \eta\left((1-\alpha)\,\|\boldsymbol{\beta}(\boldsymbol{x}_i)\|_2^2 + \alpha\,\|\boldsymbol{\beta}(\boldsymbol{x}_i)\|_1\right), \tag{3.1}$$

with regularisation parameters $\eta \geq 0$ and $\alpha \in [0,1]$. For the regularisation (3.1) to be sensible we need to ensure that all components of $\beta_j(\boldsymbol{x})$ live on the same scale. This is achieved by normalising the feature components to mean zero and unit variance.

We remark that there is a fundamental difference between the LassoNet (1.2)–(1.3) and our regularisation proposal (3.1). The LassoNet acts on the weights of the regression function, whereas we apply regularisation to the network output activations $\boldsymbol{\beta}(\boldsymbol{x}) = \Psi_W(\boldsymbol{x})$. This is called activation regularisation and was introduced in the context of recurrent neural networks by Merity et al. (2017).

We can choose any other reasonable penalty term in (3.1), e.g., if the features $\boldsymbol{x}$ naturally split into K sub-groups $\boldsymbol{x} = (\boldsymbol{x}_1^\top, \ldots, \boldsymbol{x}_K^\top)^\top \in \mathbb{R}^q$ we can also use a group LASSO penalisation, see Yuan and Li (2006),

$$\underset{W,\beta_0}{\arg\min} \frac{1}{n}\sum_{i=1}^{n} L\Big(Y_i, \mu_{W,\beta_0}(\boldsymbol{x}_i)\Big) + \sum_{k=1}^{K} \eta_k\,\|\boldsymbol{\beta}_k(\boldsymbol{x}_i)\|_2, \tag{3.2}$$

with regularisation parameters $\eta_k \geq 0$, and with $\boldsymbol{\beta}_k$ collecting the components β_j that belong to the k-th sub-group of $\boldsymbol{x}$. A group LASSO penalisation is especially suitable for categorical feature components. Of course, any other regularisation such as the fused LASSO of Tibshirani et al. (2005) can be implemented in complete analogy.

3.2 Parameter estimation

The general purpose tool for parameter estimation of (W,β_0) is the gradient descent algorithm, or variants thereof. In our application we choose a differentiable loss function L and we select the hyperbolic tangent function for the activation function ϕ in the FNN network Ψ_W, since we have found that this often performs well in modelling tabular data. This choice implies that $\boldsymbol{\beta}(\boldsymbol{x})$ and $\mu_{W,\beta_0}(\boldsymbol{x})$ are differentiable w.r.t. the parameters (W,β_0). However, other activation functions such as the ReLU function can also be used. Note that the ReLU function is differentiable almost everywhere, which allows us to apply the following considerations except on a Lebesgue null set.

The only critical term in gradient descent is the penalisation (3.1) and (3.2), respectively, and this is exactly the crucial part for variable selection. The ridge regularisation term

$$\| \boldsymbol{\beta}(\boldsymbol{x}_i) \|_2^2 = \sum_{j=1}^{q} \beta_j(\boldsymbol{x}_i)^2$$

will not cause any difficulties because it is differentiable in W. However, the LASSO term will cause difficulties because it is not differentiable in 0

$$\| \boldsymbol{\beta}(\boldsymbol{x}_i) \|_1 = \sum_{j=1}^{q} | \beta_j(\boldsymbol{x}_i) |.$$

It is exactly this non-differentiability that lets the components shrink exactly to 0 when increasing the regularisation parameter $\eta \geq 0$. Parameter estimation in LASSO regression is typically done by the proximal gradient descent algorithm which iterates an unrestricted gradient descent step with a soft-thresholding projection, mapping the unrestricted gradient descent solution back to the parameter domain; we refer to Section 5.3.3 in Hastie et al. (2015) and Parikh and Boyd (2013). The philosophy behind this is that the penalised minimisation (3.2) has a dual problem which is an optimisation restricted to a (convex) parameter domain.

Unfortunately, this soft-thresholding approach does not translate to our case. We can solve the unrestricted optimisation problem which provides us with parameter estimates $\widetilde{W}$ and $\widetilde{\beta}_0$. This gives us the regression attention estimates $\widetilde{\boldsymbol{\beta}}(\boldsymbol{x}) = \Psi_{\widetilde{W}}(\boldsymbol{x})$. The soft-thresholding operator could be used to map $\widetilde{\boldsymbol{\beta}}(\boldsymbol{x})$ back to the ℓ_1-ball, however, this is not helpful, here, as it does not tell us how to transform weight estimates $\widetilde{W}$ to meet the side constraint. Following Oelker and Tutz (2017) we choose a smooth $\varepsilon > 0$ approximation to the ℓ_1-norm. We start from a real-valued $\beta \in \mathbb{R}$ and consider

$$\| \beta \|_{2,\varepsilon} = \sqrt{\beta^2 + \varepsilon} \quad \to \quad \| \beta \|_1 = | \beta | \quad \text{as } \varepsilon \downarrow 0.$$

Thus, we can approximate $\| \beta \|_1 = | \beta |$ by $\| \beta \|_{2,\varepsilon}$ which is differentiable in 0 for any $\varepsilon > 0$. This is illustrated in Figure 1 (lhs). This is generalised to $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_q)^\top \in \mathbb{R}^q$ by setting

$$\| \boldsymbol{\beta} \|_{2,\varepsilon} = \sum_{j=1}^{q} \| \beta_j \|_{2,\varepsilon} \quad \to \quad \| \boldsymbol{\beta} \|_1 \quad \text{as } \varepsilon \downarrow 0. \tag{3.3}$$

Figure 1 (rhs) shows these approximations for $q = 2$ and $\varepsilon \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$. For the latter two ε's we can hardly see any difference to the ℓ_1-norm.

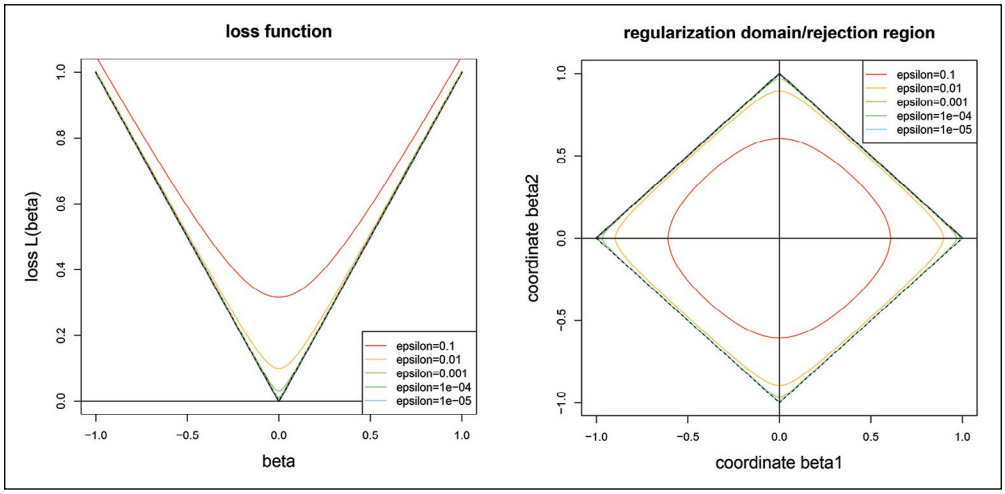

FIGURE 1 (lhs) Approximation of $\| \beta \|_1$ with $\| \beta \|_{2,\varepsilon}$ and (rhs) approximation of $\| \boldsymbol{\beta} \|_1$ with $\| \beta \|_{2,\varepsilon}$ for $q = 2$ and $\varepsilon \in \{10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}\}$.

Analogously, we can get an ε-approximation in the group LASSO case (3.2) for each term $\boldsymbol{\beta}_k(\boldsymbol{x}_i)$ of the regularisation part (with a slight abuse of notation)

$$\| \boldsymbol{\beta}_k(\boldsymbol{x}_i) \|_{2,\varepsilon} = \sqrt{\boldsymbol{\beta}_k(\boldsymbol{x}_i)^\top \boldsymbol{\beta}_k(\boldsymbol{x}_i) + \varepsilon} \quad \to \quad \| \boldsymbol{\beta}_k(\boldsymbol{x}_i) \|_2 \quad \text{as } \varepsilon \downarrow 0. \tag{3.4}$$

The disadvantage of the ε-approximations is that it does not shrink $\beta_j(\boldsymbol{x})$ exactly to zero, therefore, we still need to identify the (small) components j that should be set manually to zero. After this we have to re-run the fitting algorithm again, without considering these components. Note that such a refitting is necessary anyway because the ℓ_1 -regularisation introduces a (small) bias that should be taken care off, see e.g. Section 4.4 in Lemhadri et al. (2021) and Lee et al. (2016).

Thus, our proposal is to use the regularised LocalGLMnet in one of two different ways, depending on the problem at hand. If the aim of applying the regularised LocalGLMnet is to identify completely erroneous or irrelevant variables (for example, where some variables may have been corrupted) then the model should be fit with regularisation (3.3) for several values of the regularisation parameters η (and a small value for ε). This allows for the identification of those feature components that appear to be less important by inspecting

the regression attention values produced for each value of η. Erroneous or irrelevant variables usually display regression attention values that shrink to approximately zero once a minimal regularisation has been applied; see the examples in the following sections where fake data have been added to some of the datasets. Once these features have been identified, these can be removed and the model is then refit without using the penalised minimisation, i.e., we only utilise penalised minimisation as a feature selection tool and rely on other methods, for example, early stopping to regularise the models on the selected features. Notably, this first use of the regularised LocalGLMnet does not require the data to be split into validation or test sets, as we demonstrate on the Boston housing dataset. On the other hand, for predictive modelling of data where the aim is to produce parsimonious (i.e., sparse) and explainable models, then a different use of the regularised LocalGLMnet is appropriate. In this second case, as before, the model should be fit with regularisation (3.3) for several values of the regularisation parameters η (and a small value $\varepsilon > 0$), and the optimal value of η is identified by using a validation set. Once the optimal value has been identified, either the regularised model can be used to make predictions, or, in a second step, an optimal threshold of the average absolute regression attentions $\beta_j^{(\eta)}(\boldsymbol{x}_i)$ (see (4.3), below) can be estimated and all variables with regression attention values below this threshold can be excluded, and then a new model can be fit to the reduced dataset.

3.3 Feature pre-processing

Application of regularisation (3.1) and (3.2), respectively, requires that all components of the regression attention $\boldsymbol{\beta}(\boldsymbol{x}) \in \mathbb{R}^q$ live on a comparable scale. For components belonging to continuous variables this comparable scale is achieved by normalising the continuous feature components x_j to be centred and having unit variance. Thus, starting from raw feature components $x_{i,j}^{\text{raw}}$ of instances i we normalise

$$x_{i,j}^{\text{raw}} \mapsto x_{i,j} = \frac{x_{i,j}^{\text{raw}} - \text{mean}(x_{i,j}^{\text{raw}})_{1 \le i \le n}}{\text{stddev}(x_{i,j}^{\text{raw}})_{1 \le i \le n}}, \tag{3.5}$$

where the empirical mean and the empirical standard deviation (stddev) are calculated for fixed compone*nt j o*ver all available instances $1 \le i \le n$.

Remark that for the LocalGLMnet we typically require $x_{i,j} \neq 0$ which is almost surely the case if we scale the continuous real data according (3.5). The reason for requiring this is that otherwise the regression attention $\beta_j(\boldsymbol{x}_i)$ may not be well-specified as $\beta_j(\boldsymbol{x}_i)x_{i,j} = 0$ for $x_{i,j} = 0$ and for any $\beta_j(\boldsymbol{x}_i) \in \mathbb{R}$.

Categorical (nominal) variables are more difficult in pre-processing because we have various different options like one-hot encoding, dummy coding or Helmert's contrast coding. We propose a scaled version of dummy coding. Dummy coding leads to full rank design matrices. If we would use one-hot encoding we would have an identifiability issue in (2.2), as the intercept β_0 could be integrated into the one-hot encoded categorical variable. However, we do not want this because the intercept should be excluded from regularisation.

Assume that x_k^{raw} is a categorical variable that takes $q_k + 1 \geq 2$ levels denoted by $a_1, \ldots, a_{q_k+1}$. If we choose the first level a_1 as reference level, dummy coding of x_k^{raw} is obtained by

$$x_k^{\text{raw}} \mapsto \tilde{\boldsymbol{x}}_k = \left(\mathbb{1}_{\{x_k^{\text{raw}} = a_2\}}, \ldots, \mathbb{1}_{\{x_k^{\text{raw}} = a_{q_k+1}\}} \right)^\top \in \{0,1\}^{q_k} \subset \mathbb{R}^{q_k}. \tag{3.6}$$

This dummy coded vector $\tilde{\boldsymbol{x}}_k \in \{0,1\}^{q_k} \subset \mathbb{R}^{q_k}$ has several components equal to 0 and it is not normalised; the former again may imply difficulties in regression attentions as feature values equal to 0 leave the corresponding regression attention components undefined. For these two reasons we additionally apply standardisation (3.5) to each component of $\tilde{\boldsymbol{x}}_{i,k}$ to receive the normalised dummy coded feature components $\boldsymbol{x}_{i,k} \in \mathbb{R}^{q_k}$ for all instances $1 \leq i \leq n$. Naturally, we should apply group LASSO regularisation (3.2) and (3.4), respectively, to $\boldsymbol{x}_k \in \mathbb{R}^{q_k}$ because these feature components form a natural group of size q_k; more details on this are provided below in Section 4.2.

3.4 Comparison of methods

Above, we have defined the LocalGLMnet with regularisation which maintains the interpretable structure of the LocalGLMnet (2.2), while adding a regularisation component. On the other hand, the LassoNet (1.2)–(1.3) is more difficult to interpret since predictions are made using both a linear model component as well as a neural network; the former component is as interpretable as our proposal, whereas the latter component is not. Moreover, our proposal is less complex, requiring the choice of only a single hyper-parameter governing the extent of the regulation, η whereas LassoNet requires the choice of both a regularisation parameter η as well as a budget parameter M. Our proposal is easy to implement in state-of-the-art deep learning libraries, relying only on the usual gradient descent procedure, whereas the LassoNet requires the addition of a proximal gradient descent algorithm to this procedure (that is not supported "out of the box"). A more subtle difference between the proposals relates to the LassoNet regularisation (1.2) which only allows feature component j to enter the FFN network component to the extent that the linear regression coefficient β_j is non-zero. However, it is conceivable that some features act only in a non-linear fashion, or together with other features (as interactions). In these cases when the linear component is not as significant as the non-linear component, fitting the LassoNet likely will force the non-linear coefficients of the first hidden layer $W_j^{(1)}$ of the FFN network component of the LassoNet $\Psi_W(\boldsymbol{x})$ to be non-zero, such that the non-linearities will be captured appropriately via the FFN network. However, to accomplish this, the linear component $\boldsymbol{\beta}^\top \boldsymbol{x}$ will be forced to be non-zero and the linear coefficients $\boldsymbol{\beta}$ will probably be biased and difficult to interpret in isolation from the FFN network, since the features do not actually act in a linear fashion. Thus, in these instances, the LassoNet will likely be able to fit the data but at the cost of potential bias and a loss of interpretability.

On the other hand, our proposal does not have this limitation, and, in fact, LocalGLMnets have the universal approximation property which is common for increasing families of FFN network architectures. The final advantage of our proposal is that it is quite simple to add different types of regularisation to the model, for example, above we have considered the case of group LASSO regularisation. On the other hand, modifying the proximal gradient descent algorithm of the LassoNet is more sophisticated. In the real data examples that follow, we compare the results of the two proposals. We illustrate the methods discussed in the preceding sections in Figures 2–4, which show diagrams of a FFN network, the LassoNet and a regularised LocalGLMnet, respectively.

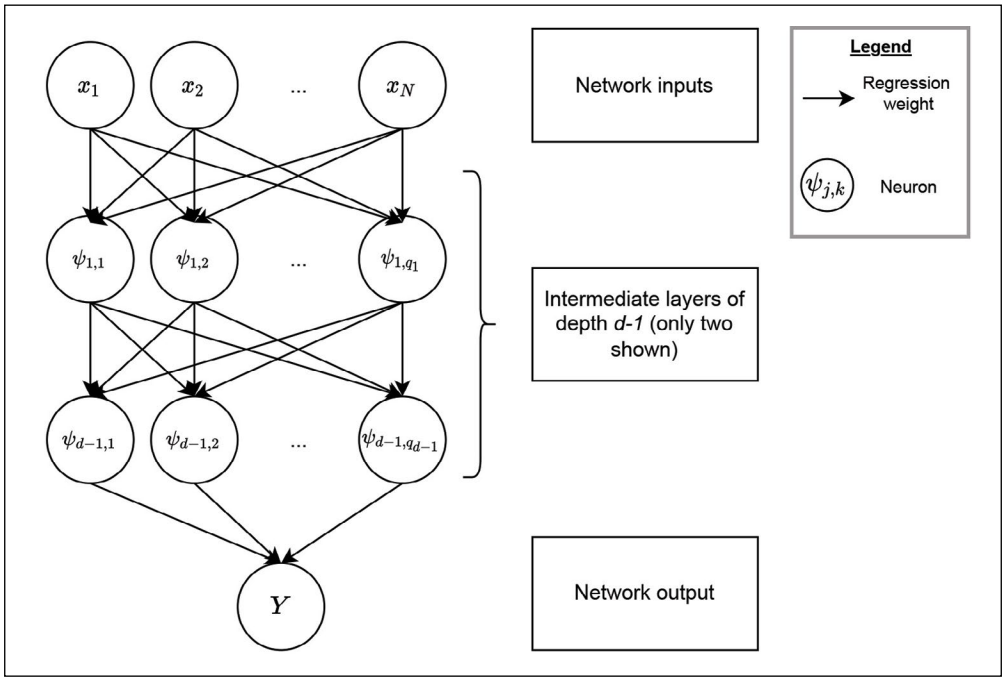

FIGURE 2 Representative diagram of a feed-forward neural (FFN) network

4. SYNTHETIC EXAMPLES

In this section we provide two toy examples illustrating the application of the proposed method in a case where the true data generating process is known; in the following section we focus on real data examples. Illustrative R code for the implementation is showcased in Appendix A.

4.1 Synthetic data example 1: LASSO regularisation

We start by considering the synthetic data example studied in Richman and Wüthrich (2022). Choose $q = 8$ feature components $\boldsymbol{x} = (x_1, \ldots, x_8)^\top \in \mathbb{R}^8$. We generate two datasets: a learning dataset $\mathcal{L}$ and a test dataset $\mathcal{T}$. The learning data will be used for model fitting

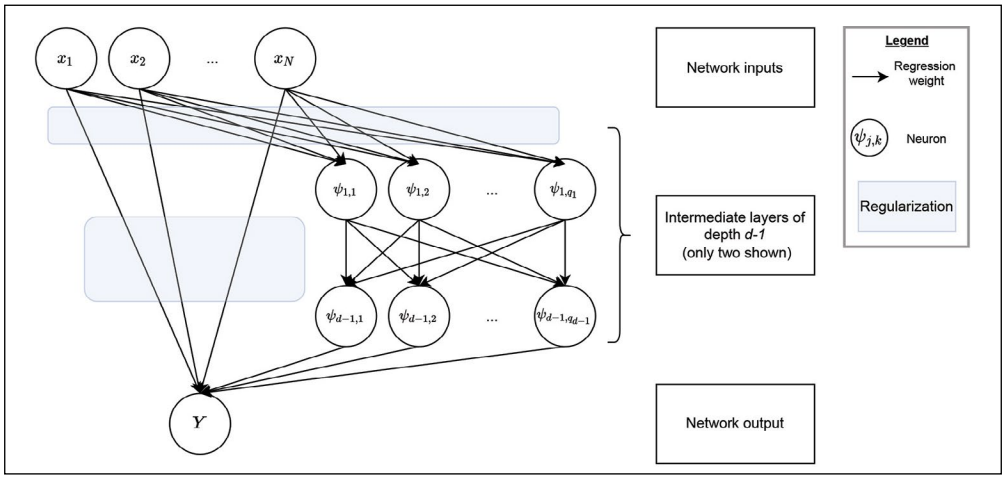

FIGURE 3 Representative diagram of a LassoNet. The shaded area in light blue represents the application of regularisation to the regression weights. Note that we have a skip connection that maps the inputs x_j directly to the output Y.

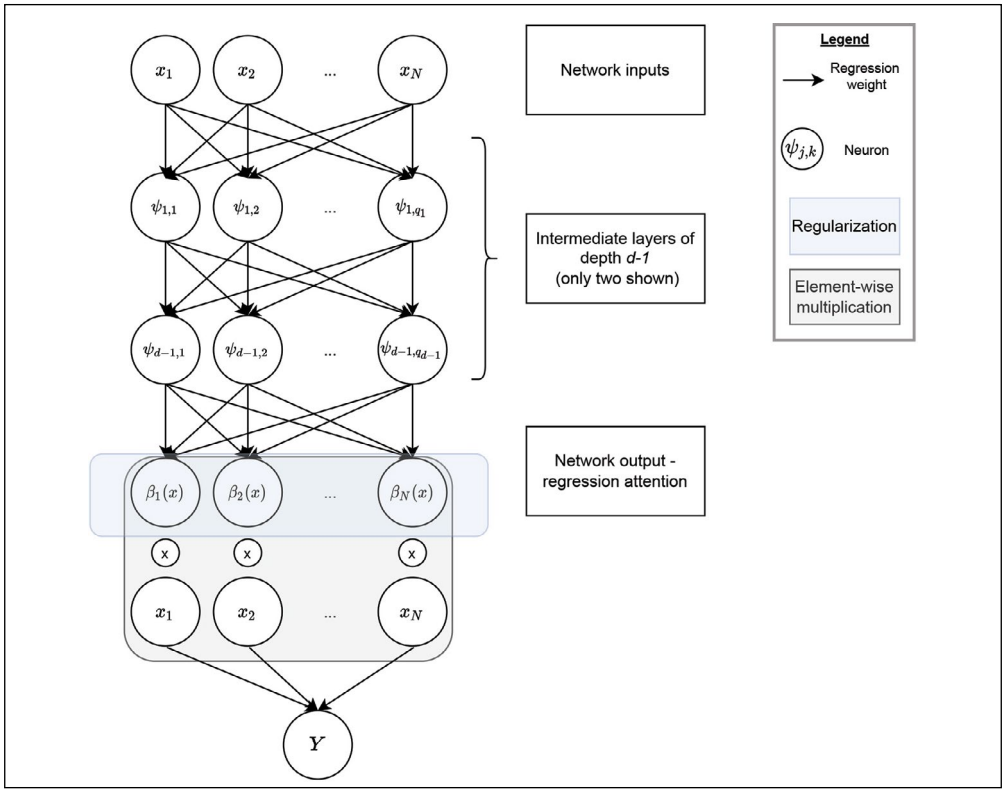

FIGURE 4 Representative diagram of a regularised LocalGLMnet. The shaded area in light blue represents the application of regularization to the regression attentions.

and the test data for an out-of-sample generalisation analysis. We choose for both datasets $n = 100{,}000$ randomly generated independent features $\boldsymbol{x} \sim \mathcal{N}(\mathbf{0}, \Sigma)$ being centred and having unit variance. Moreover, we assume that all components of $\boldsymbol{x}$ are independent, except between x_2 and x_8 we assume a correlation of 50%. Based on these features we choose the regression function

$$\boldsymbol{x} = (x_1, \ldots, x_8)^\top \in \mathbb{R}^8 \;\mapsto\; \mu^*(\boldsymbol{x}) = \frac{1}{2}x_1 - \frac{1}{4}x_2^2 + \frac{1}{2}|x_3| \sin(2x_3) + \frac{1}{2}x_4 x_5 + \frac{1}{8}x_5^2 x_6. \quad (4.1)$$

Thus, neither x_7 nor x_8 run into the (true) regression function μ^*, x_7 is independent from the remaining components, and x_8 has a 50% correlation with x_2. Based on this regression function we generate independent Gaussian observations $Y_i \sim \mathcal{N}(\mu^*(\boldsymbol{x}_i), 1)$, given $\boldsymbol{x}_i$, $i \geq 1$. This gives us the two datasets with all observations being independent

$$\mathcal{L} = \{(Y_i, \boldsymbol{x}_i); 1 \leq i \leq n\} \quad \text{and} \quad \mathcal{T} = \{(Y_t, \boldsymbol{x}_t); n+1 \leq t \leq 2n\}. \quad (4.2)$$

The goal is to determine the true regression function (4.1) from this data $\mathcal{L}$, thus, we try to approximate μ^* by a LocalGLMnet regression function μ, see (2.2). For the LocalGLMnet architecture we choose a FFN network of depth $d = 3$ with neurons $q_1 = 15$, $q_2 = 10$ and $q_3 = q = 8$, hyperbolic tangent activation function ϕ. Moreover, we choose the identity link $g(x) = x$ in (2.2). This provides us with weights (W, β_0) of dimension 384.

We fit this network on the learning data $\mathcal{L}$, for this we partition $\mathcal{L}$ at a ratio of 4:1 into training data $\mathcal{U}$ and validation data $\mathcal{V}$. We train the network on $\mathcal{U}$ and using a callback we retrieve the parameters with the lowest validation loss on $\mathcal{V}$. As loss function we choose the square loss $L(Y, \mu(\boldsymbol{x})) = (Y - \mu(\boldsymbol{x}))^2$. We choose the LASSO version $\alpha = 1$ in (3.1) with $\varepsilon = 10^{-5}$ for $\|\cdot\|_{2,\varepsilon}$, see (3.3), and we fit this model for regularisation parameters $\eta = 0, 0.125, 0.25$.

TABLE 1 Synthetic example 1: in-sample losses on $\mathcal{L}$ and out-of-sample losses on $\mathcal{T}$ of the LASSO regularised LocalGLMnet with regularisation parameters $\eta = 0, 0.125, 0.25$; the loss figures in this table only consider the square loss $\mathcal{L}$ and not the regularisation term.

	in-sample loss on $\mathcal{L}$	out-of-sample loss on $\mathcal{T}$
$\eta = 0$	1.0039	1.0106
$\eta = 0.125$	1.0238	1.0291
$\eta = 0.25$	1.0377	1.0432

Table 1 presents the results. For $\eta = 0$ we receive losses of roughly 1. This is exactly the variance of our Gaussian random variables $Y \sim \mathcal{N}(\mu^*(\boldsymbol{x}), 1)$, i.e., this is the irreducible risk. This indicates that we can find the true regression function (4.1) rather accurately in this example because we are only left with the irreducible risk. The results for $\eta > 0$ are

slightly worse, which exactly corresponds to the aforementioned bias that is induced by ℓ_1-regularisation. However, this is not the crucial point here, but we would like to see whether these regularised approaches are able to find that x_7 and x_8 should not be included into the model, see (4.1).

We consider the following simple importance measure

$$IM_j(\eta) = \frac{1}{n}\sum_{i=1}^{n}\left|\widehat{\beta}_j^{(\eta)}(\boldsymbol{x}_i)\right|, \tag{4.3}$$

for $1 \le j \le q$ and where we aggregate over all instances $1 \le i \le n$ in the learning data $\mathcal{L}$. The upper index in $\widehat{\beta}_j^{(\eta)}(\boldsymbol{x}_i)$ indicates the explicit choice of the regularisation parameter η. Generally, as explained in Remarks 2.2, we expect $IM_j(\eta)$ to be very small for the components j which only have a marginal influence on the regression function. Moreover, we expect that this importance measure is generally decreasing in η, though in general, it is not strictly decreasing.

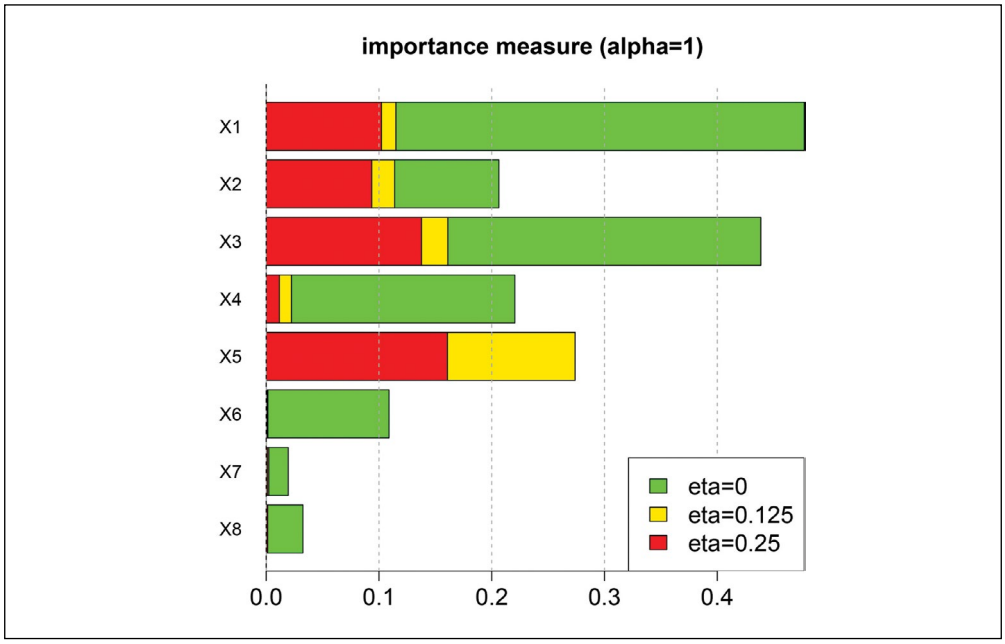

FIGURE 5 Synthetic data example 1: importance measures $IM_j(\eta)$, $1 \le j \le q = 8$, for $\eta = 0, 0.125, 0.25$.

Figure 5 shows a bar plot of the importance measures $IM_j(\eta)$, where the different colours indicate the first differences $IM_j(\eta_0) - IM_j(\eta_1)$ for $\eta_1 > \eta_0$. For the non-regularised version $\eta = 0$ the entire column lengths are relevant (red + yellow + green). We note that $IM_7(0)$ and $IM_8(0)$ are by far the smallest which doubts inclusion of the terms $\beta_7(\boldsymbol{x})x_7$ and $\beta_8(\boldsymbol{x})x_8$, respectively, into the LocalGLMnet (2.2). For (a small) LASSO regularisation $\eta = 0.125$ (red

+ yellow) we have $IM_7(0.125) \approx 0$ and $IM_8(0.125) \approx 0$ which says that we should drop the terms $\beta_7(\boldsymbol{x})x_7$ and $\beta_8(\boldsymbol{x})x_8$. Thus, we draw the right conclusions in this case.

Interestingly, also $IM_6(0.125) \approx 0$ which says that we should also drop term $\beta_6(\boldsymbol{x})x_6$. At the same time we observe $IM_5(0) \approx IM_5(0.125)$ which means that the term $\beta_5(\boldsymbol{x})x_5$ is not affected by this regularisation step. The explanation of this observation is that the LocalGLMnet is not fully identifiable, and in that case the interaction term $x_5^2 x_6 / 8$ of the true regression function (4.1) is fully allocated to $\beta_5(\boldsymbol{x})x_5$, i.e., x_5 and x_6 interact through the regression attention $\beta_5(\boldsymbol{x})$ and it absorbs the term $\beta_6(\boldsymbol{x})x_6$; see also Remarks 2.2.

We conclude that $IM_j(\eta) \approx 0$ indicates that term $\beta_j(\boldsymbol{x})x_j$ can be dropped, and one should re-run the network fitting with the dropped variable x_j to see whether the entire variable can be dropped because it does not interact with the other features, or whether only term $\beta_j(\boldsymbol{x})x_j$ is not necessary, but x_j still enters $\beta_{j'}(\boldsymbol{x})$, $j' \neq j$, in a significant way.

Table 2 Synthetic example 1: in-sample losses on $\mathcal{L}$ and out-of-sample losses on $\mathcal{T}$ without regularisation ($\eta = 0$) and dropping different sets of feature components.

	in-sample loss on $\mathcal{L}$	out-of-sample loss on $\mathcal{T}$
considering all features $\boldsymbol{x} = (x_1,\ldots,x_8)^\top$	1.0039	1.0106
considering reduced features $\boldsymbol{x} = (x_1,\ldots,x_6)^\top$	1.0101	1.0059
considering reduced features $\boldsymbol{x} = (x_1,\ldots,x_5)^\top$	1.0549	1.0506

Table 2 shows the results of the LocalGLMnet results using features reduced by (x_7, x_8) and reduced by (x_6, x_7, x_8) respectively. From these results we conclude that we can completely drop the variables (x_7, x_8), but we should keep x_6 as it has an interaction with other variables, which leads to an increase of the loss if we drop x_6; see the last line of Table 2. The fortunate result is that we draw the right conclusions for variable selection in this case. The resulting model can be illustrated and the individual terms can further be interpreted, we refer to Figures 3 and 4 in Richman and Wüthrich (2022).

4.2 Synthetic data example 2: group LASSO regularisation

The previous example only considered continuous feature components x_j, $1 \leq j \leq 8$, that were centred and with unit variance. We now extend this example by two categorical variables x_9^{raw} and x_{10}^{raw}. We choose for both variables independent categorical distributions taking the 7 labels $\{a,\ldots,g\}$ with equal probabilities of 1/7. Merging this with the Gaussian features of Section 4.1 provides us with a raw feature vector $\boldsymbol{x}^{\text{raw}} = (x_1,\ldots,x_8,x_9^{\text{raw}},x_{10}^{\text{raw}})$. Note that the first eight components are standardised because they have been generated from a centred standard Gaussian distribution, and only the categorical variables need pre-processing. We identify the labels $a_1 = a,\ldots,a_7 = g$, and we consider the following true regression function

$$\mu^*(\boldsymbol{x}^{\text{raw}}) = \frac{1}{2}x_1 - \frac{1}{4}x_2^2 + \frac{1}{2}|x_3|\sin(2x_3) + \frac{1}{2}x_4x_5 + \frac{1}{8}x_5^2x_6 + \sum_{s=1}^{7}\frac{s-4}{3}\mathbb{1}_{\{x_9^{\text{raw}}=a_s\}}. \tag{4.4}$$

Thus, neither x_7, x_8 nor x_{10}^{raw} run into the true regression function. Based on this regression function we generate independent Gaussian observations $Y \sim \mathcal{N}(\mu^*(\boldsymbol{x}^{\text{raw}}),1)$, given $\boldsymbol{x}^{\text{raw}}$, for the learning dataset $\mathcal{L}$ and the test dataset $\mathcal{T}$, see (4.2).

The goal again is to infer the true regression function μ^* from the i.i.d. learning data $(Y_i, \boldsymbol{x}_i^{\text{raw}})$, $1 \le i \le n$. In a first step we need to pre-process the two categorical variables $x_{i,9}^{\text{raw}}$ and $x_{i,10}^{\text{raw}}$ of the instances $1 \le i \le n$. Both have $q_k + 1 = 7$ levels, we choose the first level as reference level, and apply dummy coding (3.6) with $q_k = 6$ and standardisation (3.5) to each component of the dummy coded vectors. This provides us with normalised feature components $\boldsymbol{x}_{i,9}, \boldsymbol{x}_{i,10} \in \mathbb{R}^{q_k}$. Now, we are in the same situation as in Section 4.1 and we can proceed analogously. We only exchange the regularisation (3.1) by (3.2) and (3.4), respectively, because we have two groups consisting of q_k components each.

There remains the choice of the regularisation parameter $\eta_k \ge 0$ in (3.2). Following the general proposal of Yuan and Lin (2006), we scale the regularisation constants as $\eta_k = \sqrt{q_k}\,\eta$ proportionally to the square-rooted group sizes $q_k \in \mathbb{N}$. Table 3 gives the results for choices $\eta = 0, 0.05, 0.10$ and $\varepsilon = 10^{-5}$.

TABLE 3 Synthetic example 2: in-sample losses on $\mathcal{L}$ and out-of-sample losses on $\mathcal{T}$ of the group LASSO regularised LocalGLMnet fitted with parameters $\eta = 0, 0.05, 0.10$ and scaled by $\sqrt{q_k}$; the loss figures in this table only consider the square loss $\mathcal{L}$ and not the regularisation part.

	in-sample loss on $\mathcal{L}$	out-of-sample loss on $\mathcal{T}$
$\eta = 0$	1.0012	1.0059
$\eta = 0.05$	1.0143	1.0158
$\eta = 0.1$	1.1424	1.1386

Figure 6 shows a bar plot of the importance measures $IM_j(\eta)$, where the different colours again indicate the first differences $IM_j(\eta_0) - IM_j(\eta_1)$ for $\eta_1 > \eta_0$. For the two categorical variables x_9^{raw} and x_{10}^{raw} we show labels A2–A7 and B2–B7, respectively, which correspond to the levels that have not been taken as reference levels.

In the non-regularised version $\eta = 0$ (total bars including red + yellow + green) we observe that the terms for the variables x_7, x_8 and x_{10}^{raw} are the smallest, and, indeed, these are the components that do not enter the true regression function. Introducing (a small) regularisation $\eta = 0.05$ (red + yellow) shrinks these terms to zero, i.e., we come to the right conclusion. We also shrink the term for x_6 to zero, and the explanation is again the same as in the previous Section 4.1.

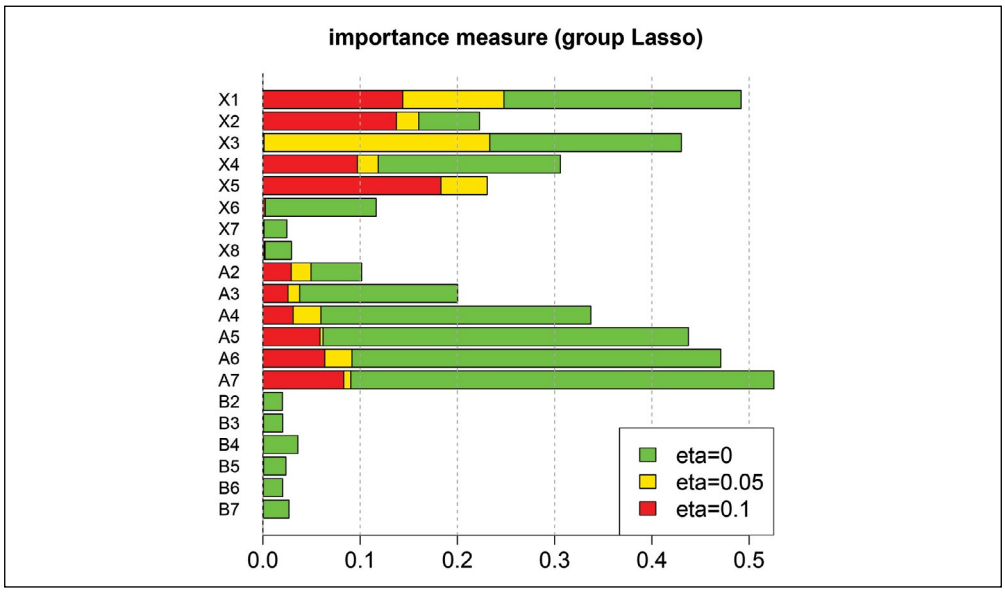

FIGURE 6 Synthetic data example 2: importance measures $IM_j(\eta)$ for $\eta = 0{,}0.05{,}0.10$.

5. REAL DATA EXAMPLES

In this section we present two applications of the regularised LocalGLMnet on publicly available real datasets of increasing complexity: the Boston housing data of Harrison and Rubinfeld (1978) and the telematics dataset of So et al. (2021). For both the Boston housing data and the telematics dataset, we compare the results of our approach to those produced with the LassoNet technique, applying the group modification of the LassoNet technique for the latter dataset.

5.1 Boston housing data

We consider the well-known Boston housing data,[2] which consists of 13 explanatory variables. They are shown in Table 7 in the appendix, and with a response variable giving the median value Y of the houses in that area. We remark that this dataset is very small, only containing $\eta = 506$ observations, and we use all of the observations for the learning data. We extend the 13 explanatory variable by 5 purely random variables X1–X5 by randomly permuting the labels of CRIM, CHAS, RM, RAD, LSTAT. This results in a feature vector $\boldsymbol{x} \in \mathbb{R}^q$ of dimension $q = 18$ from which we predict the response Y.

Firstly, we would like to see whether we can detect the purely random feature components X1–X5, and, secondly, we would like to understand whether we can drop any

2 The dataset is available at this link: http://lib.stat.cmu.edu/datasets/boston and code for this example is available on Github at this link: https://github.com/RonRichman/Regularized-LocalGLMnet

of the real feature variables, due to these being irrelevant to the problem. That is, our goal here is in line with the first use of the regularised LocalGLMnet described in Section 3.2. We only have continuous feature components, and we proceed completely analogously to Section 4.1.

From Figure 7 we conclude that we correctly discover the fake explanatory variables X1–X5. Similarly, the variable B is shrunk to zero, and this term could be removed from the dataset. While the variable DIS also appears to be less important, nonetheless this is not shrunk to zero. For a more reliable decision on this variable, we would need to apply the LocalGLMnet differently, for example, using a validation set to select η, as done in the next section.

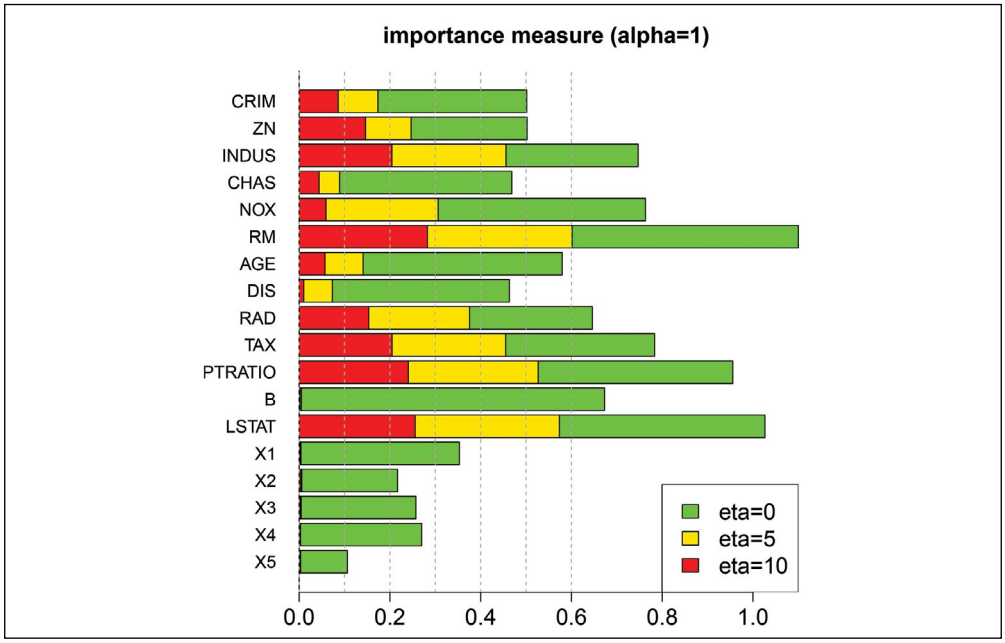

FIGURE 7 Boston housing example: importance measures $IM_j(\eta)$ for $\eta = 0,5,10$ and $\varepsilon = 10^{-5}$

Next, we consider whether applying LassoNet to the same dataset would produce similar conclusions. The LassoNet model was fit to these data using the code accompanying Lemhadri et al. (2021) as described in Appendix B. The minimum Mean Squared Error (MSE) attained on the learning data was 7.35, at a value of $\eta = 34.35$. At this value of η, all variables were included, thus, the LassoNet was not able to identify properly any of the purely random variables on this dataset. Figure 8 shows the variable importance measures produced using the LassoNet model. It can be seen that several of the real variables (ZN, TAX and INDUS) are of roughly the same magnitude as the fake variables.

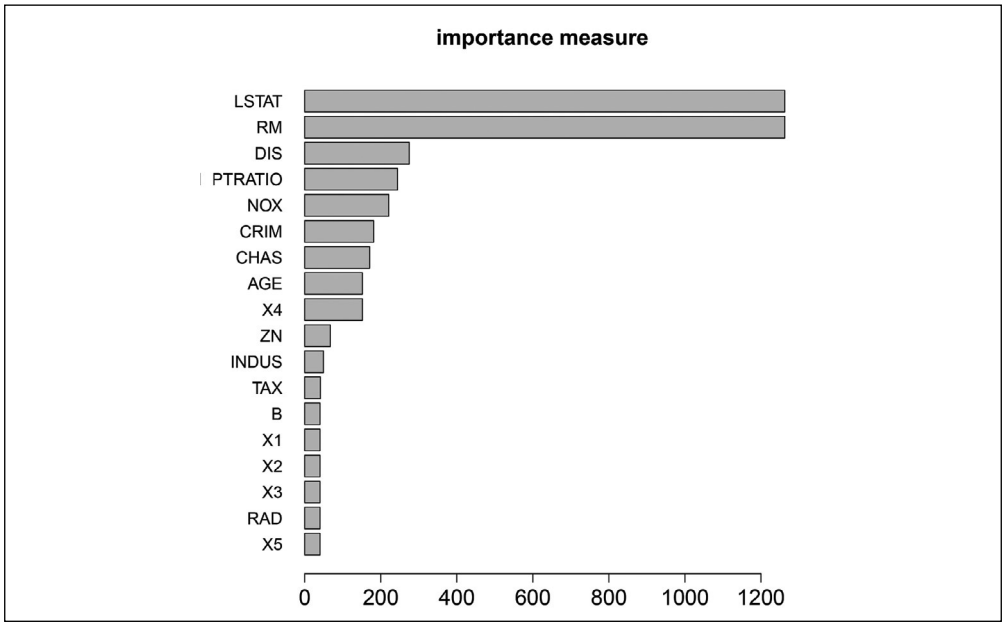

FIGURE 8 Boston housing example: importance measures produced by LassoNet.

On the other hand, Figure 9 shows the regression attentions $\beta_j(\boldsymbol{x})$ of the features in the LocalGLMnet model. It can be seen that the variables ZN, TAX and INDUS are related in a non-linear manner with the expected response variable, and, moreover, they interact significantly with other components. That is, these variables appear to play an important role in the predictions made with the LocalGLMnet model (in contrast to LassoNet). Perhaps this is because these contributions are non-linear, and the LassoNet model, which forces all variables to have a linear contribution in the first place for regularisation, may exclude these incorrectly (see discussion of this in Section 3.4).

Thus, we conclude that the regularised LocalGLMnet has been able successfully to discover the fake variables, whereas the LassoNet has not.

5.2 Telematics dataset

We consider an example with higher dimensional feature information. This is the car insurance dataset of So et al. (2021).[3] This dataset consists of 100,000 records of car insurance data from a Canadian insurer.[4] Each record contains variables relating to traditional insurance policy information which has been collected for many years within insurers, such as age and gender of the policyholder and age and usage of the insured

3 The dataset is available at: http://www2.math.uconn.edu/ valdez/telematics_syn-032021.csv

4 Note that due to privacy concerns, these 100,000 records were generated synthetically based on real data, see So et al. (2021) for a detailed description of this.

FIGURE 9 Boston housing example: regression attentions $\beta_j(\boldsymbol{x})$ learned using the LocalGLMnet. Colours of each point represent the median value (MEDV) of homes in that area.

vehicle. In addition, the records also contain variables derived from data collected via telematics devices in the insured vehicles. These telematics data have been used to produce summary statistics (variables) appearing in each record such as the annual miles driven and the frequency of harsh braking and acceleration events per 1000 miles driven (in other words, in this dataset, the raw output from the telematics devices has been processed and summarised). The variables used from this dataset are shown in Table 8 in Appendix B.

A well-known concern in the insurance industry is that protected characteristics of policyholders, such as gender in the European Union (which is legally forbidden for use in insurance pricing), can be inferred indirectly from complex high-dimensional data such as those presented in the telematics dataset, see, e.g., Lindholm et al. (2022) and the references therein for an extended discussion on discrimination in insurance. Here, we explore which of the variables within the dataset are most important for predicting the gender of the policyholder by performing a variable selection using a group LASSO regularised LocalGLMnet; here our goal is similar to the second use of the regularised LocalGLMnet described in Section 3.2. Since we aim to predict a categorical outcome, the gender of the policyholder (which was coded as 0 for females and 1 for males), we slightly modify the output of the LocalGLMnet by adding a sigmoid activation to the output of the network so that the estimates are constrained to lie in the range (0,1). Note that the sigmoid activation function is the inverse of the canonical link of the logistic regression model. This motivates us to use a regularised version of the binary cross-entropy loss (Bernoulli deviance loss) for the loss function. This binary cross-entropy loss is for the binary response Y given by

$$L(Y, \mu(\boldsymbol{x})) = -Y \log(\mu(\boldsymbol{x})) - (1-Y)\log(1-\mu(\boldsymbol{x})).$$

This loss is used in (3.2).

We choose a FFN network component for the LocalGLMnet architecture being structured with three hidden layers, having $q_1 = 45$, $q_2 = 20$ and $q_3 = q = 24$ hidden neurons; note that a slightly more complex FFN network is used compared to the previous examples given the expected non-linearities in these telematics data. Networks were fit for the values of the regularisation parameter of $\eta = 0, 3.3\times10^{-4}, 6.6\times10^{-4}, 1\times10^{-3}, 3.3\times10^{-3}, 6.6\times10^{-3}, 1\times10^{-2}, 3.3\times10^{-2}, 6.6\times10^{-2}, 1\times10^{-1}$, and we set $\varepsilon = 10^{-5}$. To choose the optimal regularisation parameter η for a good predictive accuracy, we split the 100,000 observations into three disjoint sets: first we partition into a learning set $\mathcal{L}$ consisting of 90% of the records and a test set $\mathcal{T}$ of 10%. Subsequently, we split the learning set $\mathcal{L}$ further into a validation set $\mathcal{V}$ consisting of 10% of the learning set $\mathcal{L}$ and a training set $\mathcal{U}$ of the remaining 90% of the learning set $\mathcal{L}$. The networks were fit for 50 epochs using a batch size of 128 on the training data $\mathcal{U}$ and the validation data $\mathcal{V}$ is used for the selection of the regularisation parameter η.

Figure 10 shows a bar plot of the importance measures $IM_j(\eta)$, for the telematics dataset, with the bars ordered by the importance measure at the highest level of regularisation, i.e.

$\eta = 0.1$. In this plot, we have excluded the territory variable, since, overall, the variable importance measure for the different levels of this variable is low; see Figure 12 in Appendix B for these. It can be observed that the variables with the highest importance measures relate to a mixture of traditional and telematics variables. Of the latter, the most important of these relate to harsh braking and acceleration events, the total distance that a vehicle is driven each year, and the extent to which the vehicle usage occurs on specific days and day times. The least important of the telematics variables relate to high intensity turning events, see Figure 10.

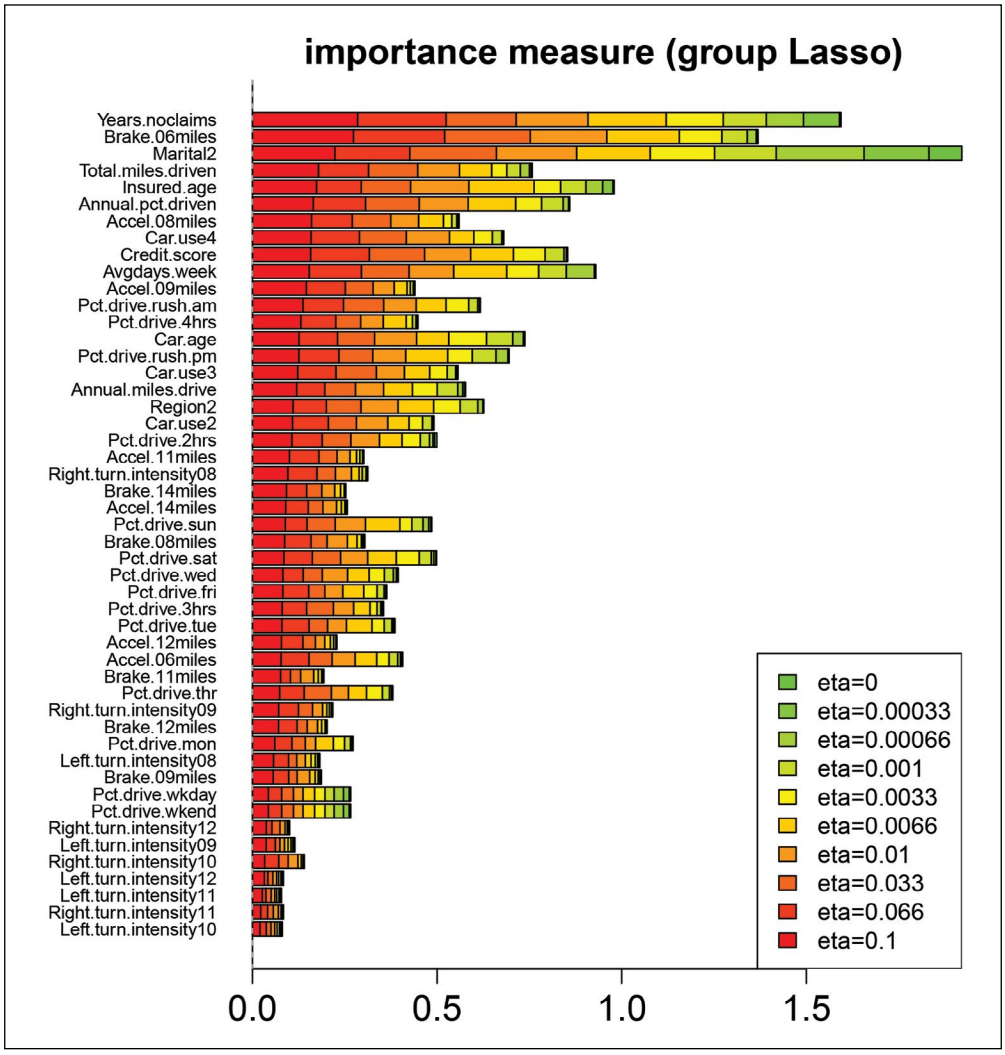

FIGURE 10 Telematics example: importance measures produced by the group LASSO regularised LocalGLMnet; validation set. Note that the territory variable has been excluded.

In Table 4, we show the binary cross-entropy losses achieved on the training set $\mathcal{U}$ and validation set $\mathcal{V}$, with $\eta = 0.0033$ producing the lowest loss on the validation set $\mathcal{V}$, thus, in what follows, we select the LocalGLMnet fit with this level of regularisation.

Table 4 Telematics example: training set losses on $\mathcal{U}$ and validation set losses on $\mathcal{V}$ of the group LASSO regularised LocalGLMnet.

	training loss on $\mathcal{U}$	**validation loss on $\mathcal{V}$**
$\eta = 0$	0.6159	0.6603
$\eta = 0.00033$	0.6267	0.6641
$\eta = 0.00066$	0.6285	0.6543
$\eta = 0.001$	0.6356	0.6645
$\eta = 0.0033$	**0.6159**	**0.6539**
$\eta = 0.0066$	0.6372	0.6608
$\eta = 0.01$	0.6465	0.6576
$\eta = 0.033$	0.6335	0.6605
$\eta = 0.066$	0.6466	0.6638
$\eta = 0.1$	0.6714	0.6758

On the test set $\mathcal{T}$ (which has not been used, yet), the results produced using this regularised LocalGLMnet, as well as an unregularised LocalGLMnet and a regular FFN network, are shown on rows 1–3 of Table 5. It can be seen that, among these models, the regularised LocalGLMnet performs the best here, followed by the FFN network and then the regular LocalGLMnet. Thus, on this example of telematics data, the regularised LocalGLMnet performs better than the FFN network (out-of-sample), and additionally it provides a significantly more explainable prediction than the FFN network.

Table 5 Telematics example: test set losses on $\mathcal{T}$ of the group LASSO regularised LocalGLMnet compared to other approaches.

	test loss on $\mathcal{T}$
Regularised LocalGLMnet	0.6520
LocalGLMnet	0.6637
FFN network	0.6622
LocalGLMnet - reduced data	0.6448
FFN network - reduced data	0.6478
LassoNet	0.6604

In a final step, we proceed to perform variable selection. We choose a threshold value for the importance measures $IM_j(\eta = 0.0033)$, $1 \leq j \leq q$, beneath which we consider that the

variables should be excluded from the LocalGLMnet. We proceed by estimating the deciles of the importance measures produced by the regularised LocalGLMnet, and successively set to zero those variables with an importance measure less than each decile. At each step, we refit the model and re-estimate the cross-entropy loss on the validation set $\mathcal{V}$. The results of this procedure are shown in Table 6, where it can be seen that the lowest validation loss is produced by setting all variables with an importance measure less than the median to zero. At the median importance measure, the majority of the territory variables have been set to zero, and, heuristically, for the final set of selected variables, all of the territory variables were set to zero (similarly to the functioning of the group LASSO). The final set of selected variables for this model is shown in Figure 11, which has been reduced from 103 variables to 32.

Table 6 Telematics data example: validation set losses on $\mathcal{V}$ of the group LASSO regularised LocalGLMnet with a regularisation parameter of $\eta = 0.0033$ as successive variable selection is performed.

deciles	value of IM_j at decile	validation loss on $\mathcal{V}$
0.1	0.0131	0.6554
0.2	0.0163	0.6575
0.3	0.0181	0.6560
0.4	0.0199	0.6580
0.5	0.0222	0.6546
0.6	0.0249	0.6553
0.7	0.0291	0.6577
0.8	0.0601	0.6684
0.9	0.0873	0.6772
1	0.2118	0.7087

We conclude by refitting the LocalGLMnet and a FFN network on this reduced set of variables. These results are shown on rows 4–5 of Table 5. Reducing the variables on which the models were fit has improved the losses of both models significantly compared to the regularised LocalGLMnet, and, in this case, the FFN network is slightly outperformed by the LocalGLMnet. This outperformance of the LocalGLMnet compared to the regular FFN network is likely because the variable selection has been performed in the context of the LocalGLMnet structure, which uses the selected variables both as inputs to the FFN network component as well as multiplying these by the output of the FFN network component (which are the regression attentions). In other words, the threshold was not chosen using a regular FFN network but rather the selected threshold is optimal for the LocalGLMnet. The final row of Table 5 shows the result of fitting a LassoNet model, see

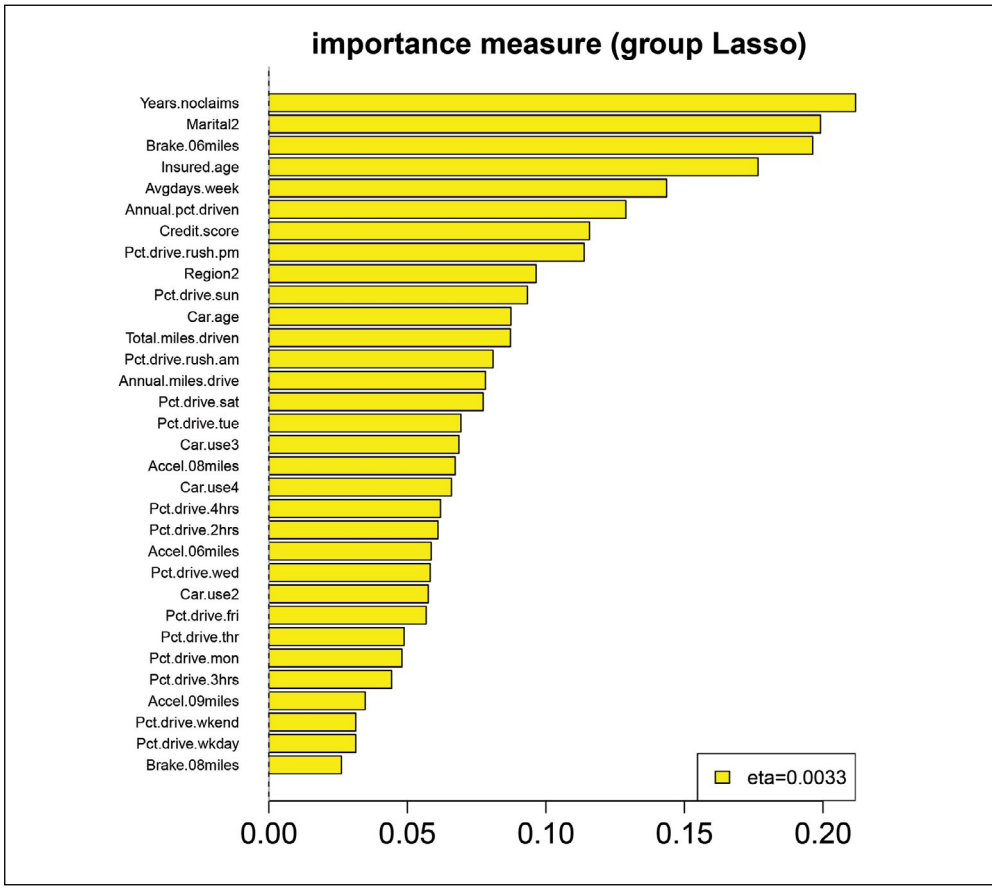

FIGURE 11 Telematics example: final selected variables with regularisation parameter $\eta = 0.0033$ and median importance measure score

Appendix B for more details on how the model was trained. The training process led to the selection of a model with no regularisation, thus, no variables were excluded. The LassoNet performs better than the non-regularised LocalGLMnet as well as the FFN network, here. However, the regularised LocalGLMnet and the models fit on the reduced dataset outperform the LassoNet.

6. CONCLUSIONS

The LocalGLMnet provides an interpretable deep learning model benefiting from a generalised additive decomposition that allows for variable selection. In our original proposal (Richman and Wüthrich, 2022) we have proposed an empirical Wald test to perform variable selection for continuous feature variables, and categorical features have been excluded from this proposal. In the present paper, we extend the LocalGLMnet proposal by a regularisation term during model fitting. Basically, any reasonable

regularisation function can be chosen; however, for enforcing feature sparsity the group LASSO proposal is the most suitable one as it allows for variable selection also among grouped variables such as categorical variables (with dummy coding). In our examples, we have seen that this variable selection proposal works properly. This has been done in two different ways. First, in the context of synthetic data examples where the ground truth was known, as well as on real data examples. We showed that the variable selection is able to identify fake variables added to the data. Second, we have shown how the regularised LocalGLMnet can be applied in the context of variable selection and modelling of a complex dataset where the importance of each variable is not known at the outset. In this example, the regularised LocalGLMnet outperforms a plain vanilla feed-forward neural network. Moreover, our proposal has been benchmarked with the LassoNet of Lemhadri et al. (2021), and at least in the examples studied we give preference to our LASSO regularised LocalGLMnet approach. Nonetheless, we note that relatively little hyperparameter tuning was performed on the LassoNet, and it is likely that with a more thorough tuning, the results of the LassoNet could improve.

The LocalGLMnet is more complex than a plain vanilla feed-forward neural network, and it might suffer from slightly less predictive power because of a faster in-sample over-fitting in gradient descent fitting. If this is the case, we have proposed and demonstrated a two-stage approach, namely, first perform variable selection within a group LASSO regularised LocalGLMnet architecture, and, second, fit a plain vanilla feed-forward neural network on the selected variables. We remark that a second fitting loop should be done in any case because LASSO regularisation likely leads to a biased model – see Section 4.4 in Lemhadri et al. (2021) and Lee et al. (2016).

ACKNOWLEDGEMENT

The authors wish to thank the editor, assistant editor and reviewers of an earlier version of this manuscript for their comments which helped to improve the manuscript significantly.

REFERENCES

Agarwal, R, Frosst, N, Zhang, X, Caruana, R & Hinton, GE (2020). Neural additive models: interpretable machine learning with neural nets. *arXiv*:2004.13912v1.

Apley, DW & Zhu, J (2020). Visualizing the effects of predictor variables in black box supervised learning models. *Journal of the Royal Statistical Society: Series B*, **82**(4): 1059–1086.

Friedman, JH (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, **29**(5): 1189–1232.

Gneiting, T (2011). Making and evaluating point forecasts. *Journal of the American Statistical Association*, **106**(494): 746–762.

Gneiting, T & Raftery, AE (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, **102**(477): 359–378.

Harrison, D & Rubinfeld, DL (1978). Hedonic prices and the demand for clean air. *Journal of Environmental Economics & Management*, **5**, 81–102.

Hastie, T, Tibshirani, R & Wainwright, M (2015). *Statistical Learning with Sparsity: The Lasso and Generalizations.* CRC Press.

Hoerl, A & Kennard, R (1970). Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, **12**, 55–67.

Lee, JD, Sun, DL, Sun, Y & Taylor, JE (2016). Exact post-selection inference, with application to the LASSO *Annals of Statistics*, **44**(3): 907–027.

Lemhadri, I, Ruan, F, Abraham, L & Tibshirani, R (2021). LassoNet: a neural network with feature sparsity. *Journal of Machine Learning Research*, **22**, 1–29.

Lindholm, M, Richman, R, Tsanakas, A & Wüthrich, MV (2022). Discrimination-free insurance pricing. *ASTIN Bulletin*, **52**, 55–89.

Merity, S, McCann, B & Socher, R (2017). Revisiting activation regularization for language RNNs. *arXiv*:1708.01009v1.

Merz, M, Richman, R, Tsanakas, A & Wüthrich, MV (2022). Interpreting deep learning models with marginal attribution by conditioning on quantiles. *Data Mining and Knowledge Discovery*, in press.

Oelker, M-R & Tutz, G (2017). A uniform framework for the combination of penalties in generalized structured models. *Advances in Data Analysis and Classification*, **11**, 97–120.

Parikh, N & Boyd, S (2013). Proximal algorithms. *Foundations and Trends in Optimization*, **1**(3): 123–231.

Richman, R (2021). Mind the gap – safely incorporating deep learning models into the actuarial toolkit. *SSRN Manuscript* ID 3857693.

Richman, R & Wüthrich, MV (2022). LocalGLMnet: interpretable deep learning for tabular data. *Scandinavian Actuarial Journal*, in press.

So, B, Boucher, JP & Valdez, EA (2021). Synthetic dataset generation of driver telematics. *Risks*, **9**(4): 58.

Tibshirani, R (1996). Regression shrinkage and selection via the LASSO. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, **58**, 267–288.

Tibshirani, R, Saunders, M, Rosset, S, Zhu, J & Knight, K (2005). Sparsity and smoothness via the fused LASSO. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, **67**, 91–108.

Tikhonov, AN (1943). On the stability of inverse problems. *Doklady Akademii Nauk SSSR*, **39**(5): 195–198.

Vaswani, A, Shazeer, N, Parmar, N, Uszkoreit, J, Jones, L, Gomez, AN, Kaiser, Ł & Polosukhin, I (2017). Attention is all you need. *arXiv*:1706.03762v5.

Vaughan, J, Sudjianto, A, Brahimi, E, Chen, J & Nair, VN (2018). Explainable neural networks based on additive index models. *arXiv*:1806.01933v1.

Yuan, M & Lin, Y (2006). Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, **68**, 49–67.

Zou, H & Hastie, T (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, **67**, 301–320.

APPENDIX A

R code

Listing 1 ElasticNet regularisation code

```
Design    = layer_input(shape = c(q[1]), dtype = 'float32', name = 'Design')
BiasTerm = layer_input(shape = c(1), dtype = 'float32', name = 'BiasTerm')
#
Attention = Design %>%
        layer_dense(units=q[2], activation='tanh', name='FNLayer1') %>%
        layer_dense(units=q[3], activation='tanh', name='FNLayer2') %>%
        layer_dense(units=q[1], activation='linear', name='Attention')
#
PenaltyL1 = Attention %>% layer_lambda(function(x) k_square(x)) %>%
        layer_lambda(function(x) eta*alpha*k_sqrt(x+epsilon)) %>%
        layer_dense(units=1, activation='linear', name='PenaltyL1',
            weights=list(array(1, dim=c(q[1],1))), use_bias=FALSE, trainable=FALSE)
#
PenaltyL2 = Attention %>% layer_lambda(function(x) k_square(x)) %>%
        layer_lambda(function(x) eta*(1-alpha)*x) %>%
        layer_dense(units=1, activation='linear', name='PenaltyL2',
            weights=list(array(1, dim=c(q[1],1))), use_bias=FALSE, trainable=FALSE)
#
Penalty = list(PenaltyL1,PenaltyL2) %>% layer_add()
#
LocalGLM = list(Design, Attention) %>% layer_dot(name='LocalGLM', axes=1)
#
Bias = BiasTerm %>%
        layer_dense(units=1, activation='linear', name='Bias', use_bias=FALSE)
#
Response = list(LocalGLM, Bias) %>% layer_add()
Output = list(Response, Penalty) %>% layer_concatenate()
#
model <- keras_model(inputs = c(Design, BiasTerm), outputs = c(Output))
```

Listing 2 Regularised loss

```
MSEP_regularized <- function(y_true, y_pred){
                        k_mean((y_true[,1]-y_pred[,1])^2 + y_pred[,2])}
#
model %>% compile(loss = MSEP_regularized, optimizer = 'nadam')
```

Listing 3 Group Lasso penalisation code

```
groups <- c(1,1,1,1,1,1,1,1,6,6)
#
group.lasso.grouping <- function(xx){
    gg <- array(0, dim=c(length(xx),sum(xx)))
    for (k in 1:length(xx)){
        if (k==1){gg[k,1:xx[k]] <- sqrt(xx[k])
                }else{
                gg[k,(sum(xx[1:(k-1)])+1):sum(xx[1:k])] <-  sqrt(xx[k])
                }}
    t(gg)
    }
#
gg <- group.lasso.grouping(groups)
#
Penalty = Attention %>%
            layer_lambda(function(x) k_square(x)) %>%
            layer_dense(units=length(groups), activation='linear',
                weights=list(gg), use_bias=FALSE, trainable=FALSE) %>%
            layer_lambda(function(x) eta*k_sqrt(x+epsilon)) %>%
            layer_dense(units=1, activation='linear', name='Penalty',
                weights=list(array(1, dim=c(length(groups),1))), use_bias=FALSE, trainable=FALSE)
```

APPENDIX B

LassoNet – Training details

The LassoNet models used were based on the Python code provided for the group LASSO version of the LassoNet at https://github.com/lasso-net/lassonet/tree/group-lasso.[5] The dimensions of the hidden layers of the LassoNet were set equal to the same dimensions as the corresponding LocalGLMnet so that the model capacity is roughly comparable, i.e., any differences in performance will be mainly attributable to the way in which regularisation is applied within each of the models. The main hyperparameter tested for the LassoNet was the budget parameter M; a range of LassoNet models are fit automatically for different values of the regularisation parameter η. Values of $M \in \{1,10,100\}$ were tested for each example.

For the Boston housing dataset, the best LassoNet model as indicated by the MSE on the learning set was selected (since there are no validation or test sets used in that example). For the telematics data, the LassoNet producing the lowest values of the binary cross-entropy loss on the validation set was selected.

Only a single run of the LassoNet model was used for these results. Nonetheless, it was observed that the results varied quite significantly for each training run indicating that better results could be perhaps achieved using multiple runs and averaging over these.

Table 7 Boston housing data – feature components used

variable	description
CRIM	per capita crime rate by town
ZN	proportion of residential land zoned for lots over 25,000 sq.ft.
INDUS	proportion of non-retail business acres per town
CHAS	Charles River dummy variable (1 if tract bounds river; 0 otherwise)
NOX	nitric oxides concentration (parts per 10 million)
RM	average number of rooms per dwelling
AGE	proportion of owner-occupied units built prior to 1940
DIS	weighted distances to five Boston employment centres
RAD	index of accessibility to radial highways
TAX	full-value property-tax rate per 10,000
PTRATIO	pupil-teacher ratio by town
B	$1000(Bk-0.63)^2$ where Bk is the proportion of blacks by town
LSTAT	lower status of the population
X1	randomly permuted CRIM
X2	randomly permuted CHAS
X3	randomly permuted RM
X4	randomly permuted RAD
X5	randomly permuted LSTAT

5 The grouped version of the model was applied in accordance with the instructions at https://github.com/lasso-net/lassonet/issues/7.

TABLE 8 Telematics data – feature components used

variable	description	type
Insured.age	Age of insured driver, in years	continuous
Car.age	Age of vehicle, in years	continuous
Marital	Marital status (Single/Married)	categorical
Car.use	Use of vehicle: Private, Commute, Farmer, Commercial	categorical
Credit.score	Credit score of insured driver	continuous
Region	Type of region where driver lives: rural, urban	categorical
Annual.miles.drive	Annual miles expected to be driven declared by driver	continuous
Years.noclaims	Number of years without any claims	continuous
Territory	Territorial location of vehicle	categorical
Annual.pct.driven	Annualised percentage of time on the road	continuous
Total.miles.driven	Total distance driven in miles	continuous
Pct.drive.xxx	Percent of driving day xxx of the week: mon/tue/. . . /sun	continuous
Pct.drive.xhrs	Percent vehicle driven within x hrs: 2hrs/3hrs/4hrs	continuous
Pct.drive.xxx	Percent vehicle driven during xxx: wkday/wkend	continuous
Pct.drive.rushxx	Percent of driving during xx rush hours: am/pm	continuous
Avgdays.week	Mean number of days used per week	continuous
Accel.xxmiles	Number of sudden acceleration 6/8/9/. . . /14 mph/s per 1000 miles	continuous
Brake.xxmiles	Number of sudden brakes 6/8/9/. . . /14 mph/s per 1000 miles	continuous
Left.turn.intensityxx	Number of left turn per 1000 miles with intensity 08/09/10/11/12	continuous
Right.turn.intensityxx	Number of right turn per 1000 miles with intensity 08/09/10/11/12	continuous

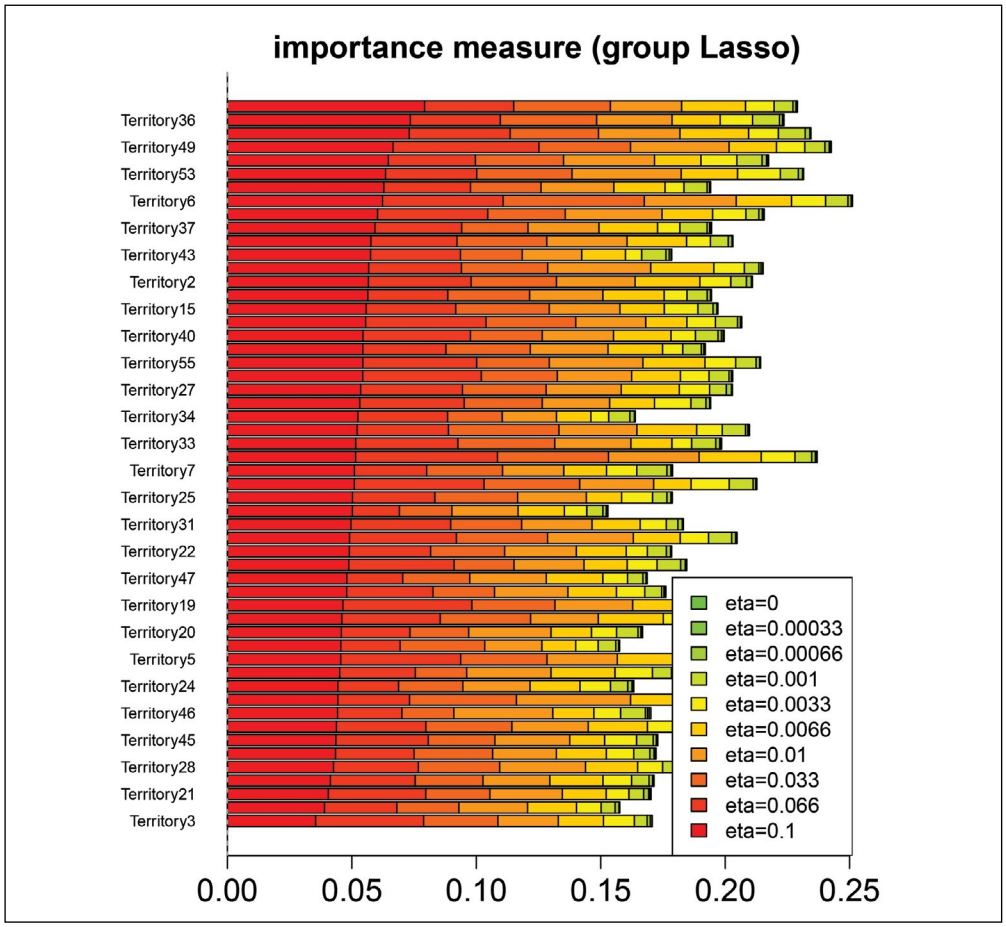

FIGURE 12 Telematics example: importance measures produced by group LASSO regularised LocalGLMnet; validation set. Territory variable only.