A Unifying Network Architecture for Semi-Structured Deep Distributional Learning

David Rügamer 1  Chris Kolb 2  Nadja Klein 2

Abstract

We propose a unifying network architecture for deep distributional learning in which entire distributions can be learned in a general framework of interpretable regression models and deep neural networks. Previous approaches that try to combine advanced statistical models and deep neural networks embed the neural network part as a predictor in an additive regression model. In contrast, our approach estimates the statistical model part within a unifying neural network by projecting the deep learning model part into the orthogonal complement of the regression model predictor. This facilitates both estimation and interpretability in high-dimensional settings. We identify appropriate default penalties that can also be treated as prior distribution assumptions in the Bayesian version of our network architecture. We consider several use-cases in experiments with synthetic data and real world applications to demonstrate the full efficacy of our approach.

1. Introduction

Machine learning (ML) approaches are able to achieve an outstanding performance in regression problems with respect to certain metrics, such as the mean squared error. However, they often cannot provide valid uncertainties on the distribution of the target variables (Song et al., 2019). To address this in regression problems, Kuleshov et al. (2018); Song et al. (2019) proposed to apply post-hoc corrections, where a pre-trained probabilistic model is taken as output in a second model, so that the result is a better calibrated distribution. Alternatively, quantile regression has been considered, however, quantile-level calibrated regressors do not ensure proper calibration for a particular prediction (unlike classification), an issue that can be overcome by distributional regression models (Rigby & Stasinopoulos, 2005). The general idea here is to extend classical mean regression models, which relate the expectation of a $M$-dimensional target variable $Y = (Y_1, \ldots, Y_M)^\top$ to input features, to models for an arbitrary parametric distribution $D(\theta_1, \ldots, \theta_K)$, with density $p_D$ provided that $\partial \log(p_D)/\partial \theta_k$ exists for all $k = 1, \ldots, K$. Since all parameters $\theta_k$ can be non-linear transformations $f$ of inputs $\nu$, i.e. $\theta_k \equiv \theta_k(f(\nu))$ one can avoid the pre-training and correction step of post-hoc approaches. In addition, this framework can easily account for skewed, count, mixed or multivariate outputs. A simple example is the univariate normal distribution where mean $(\theta_1(f_1(\nu)))$ and variance (or scale, $\theta_2(f_2(\nu))$) can be learned. On the downside distributional regression models assume structured transformations $f$, not allowing for high-dimensional interactions of features or non-tabular inputs. To extend this approach for more complex structures of tabular inputs, recent proposals try to add a single layer neural network (Umlauf et al., 2018) to a structured predictor in a regression model or permit a regression predictor in the class of generalized linear models (GLMs) that is based on a deep neural network (Tran et al., 2019).

In this paper, we propose an orthogonalization cell that allows for the joint estimation of the structured model part and the unstructured deep neural network (DNN) predictor combined as the direct sum (in a vector space meaning) in a unifying network. This leads to a new network architecture enabling the estimation of interpretable linear or non-linear effects of single features, while still allowing for higher-order interactions of those features, as well as the inclusion of non-standard inputs, such as images or text, through the DNN predictor. This approach exploits the advantages of a unified network structure similar to (Cheng et al., 2016), who focus on the application to recommender systems and GLMs. We consider distributional learning, which includes GLMs as a special case. Additionally, our approach permits uncertainty assessment using Bayesian principles and, in particular, efforts to make additive predictors identifiable, which is crucial for the interpretation of resulting models.
1.1. Main Features

Modularity: Our holistic concept of estimating the regression model within a network permits the specification of model components such as the deep network structure, the optimizer or the inference procedure independently of the distribution that is assumed or mimicked by a corresponding loss function. Furthermore, by projecting the deep network predictor into the orthogonal complement of the structured predictor and projections therein, we allow for flexible separability assumptions and corresponding constraints.

Semi-structured predictor/multimodality: Our approach encompasses non-linear relationships of features of any kind, as well as the possibility to account for non-tabular data inputs due to the deep network part. As a consequence, we technically constitute a multimodal deep learning approach (Ngiam et al., 2011), while giving researchers the freedom to define a structured model corresponding to their hypotheses.

High-dimensional regression: In contrast to similar approaches, the given framework can easily handle settings with more features than observations.

Calibration of target variables: The distributional framework considered in this paper ensures calibration of the target distribution through the rich class of potential distributions and the ability to have feature effects on all distribution parameters. Examples include skewed, (zero-inflated) count, mixed or multivariate outputs. In the latter case, fitting regression models as neural networks allows for both a joint distribution accounting for potential dependencies between outputs and any input/output association.

Improved inferential aspects: Incorporating structured additive effects into a neural network allows for more control over the influence of certain features, can improve predictive performance and makes hypothesis testing and other statistical inference procedures available in deep learning use cases. This is also in line with the results of our experiments, indicating that the inclusion of structured effects tends to be less prone to overfitting.

Automatic feature selection: The framework has an inbuilt model selection mechanism and allows researchers to separate simple linear and non-linear effects of features from the “black-box” part of the neural network, thereby also yielding an interpretable model in the spirit of interpretable machine learning (IML; Doshi-Velez & Kim, 2017).

Efficient implementation: Estimating the model within a unifying neural network enables us to take advantage of fast and well-tested machine learning platforms. We use TensorFlow (TF) and add-on module TensorFlow Probability (TFP) which allows extensions of the framework in for our purposes and offers multiple options for statistical inferential procedures (Abadi et al., 2016; Dillon et al., 2017).

2. Semi-Structured Deep Distributional Learning

The proposed architecture is a large network, consisting of several smaller networks with different input layers, layers with trainable parameters and a common distribution layer, where each parameter $\theta_1, \ldots, \theta_K$ of the distribution $\mathcal{D}$ can be modeled using a combination of smaller networks. We call our approach semi-structured deep distributional learning (SDDL) and describe its components in the remainder of this section.

2.1. The Core Architecture in a Nutshell

An exemplary architecture of an orthogonalization cell (see Section 2.4 for details) is visualized in Figure 1. The cell processes the available feature vector $\nu$ with linear input features $x = \{x_1, \ldots, x_p\}$, structured non-linear input features $z = \{z_1, \ldots, z_r\}$ and features $u = \{u_1, \ldots, u_q\}$ that are passed through a deep neural network. We call these three components structured linear inputs, structured non-linear inputs and unstructured (non-linear) inputs. Both structured input types are modeled as a single unit hidden layer with linear activation functions and different regularization terms. The unstructured deep neural network model part(s) can be arbitrarily specified. Inputs in one or more DNNs are features that are assumed to have a non-linear effect or interact with each other in a complex way. A distinction is made between DNNs whose unstructured inputs are also part of the structured inputs, i.e. $\exists u \in u : u \in x \cup z$, and DNNs whose inputs solely appear in the unstructured predictor (illustrated by an XOR-node in Figure 1). In the latter case, the DNN outputs are passed through to the Deep input gate, directly summed up with structured inputs and fed into the distributional layer. DNNs that also share inputs with $x$ or $z$ are connected to the Deep decomposed input gate, which applies an orthogonalization operation to the penultimate DNN layer before adding its outputs to the remaining predictors.

2.2. Output Model Structure

In contrast to conventional regression models only modeling the conditional mean $\mu := \mathbb{E}_{\mathcal{D}}(Y | \nu)$ of $\mathcal{D}$, distributional regression (DR) allows to regress features $\nu$ on potentially all $K$ parameters $\theta = (\theta_1, \ldots, \theta_K)^T$ of the outcome distribution. By assuming a parametric distribution $Y | \nu \sim \mathcal{D}(\theta_1(\nu), \ldots, \theta_K(\nu))$, the goal of DR is to estimate the relationship of each of the $K$ distribution parameters and the given features $\nu$ through monotonic and differentiable response functions $h_k$

$$h_k(\nu_i) = h_k(\eta_k(\nu_i)).$$

The predictors $\eta_k(\nu_i) \in \mathbb{R}$ hence specify the relationship between each feature $\nu_j$ in $\nu$ and the (transformed) param-
Semi-Structured Deep Distributional Learning

Figure 1. Exemplary architecture of an orthogonalization cell for one distribution parameter: A structured linear predictor and a structured non-linear predictor (represented by a basis evaluation) are fed into the cell and combined with the deep network using the orthogonalization operation. All three predictors are summed and the output is finally passed on to a distributional layer where it can be used for one or more parameters of a parametric distribution.

ders $h_k^{-1}(\theta_k)$, while $h_k$ ensures that possible parameter space restrictions on $\theta_k$ are fulfilled for each $\nu$.

In order to embed DR into a neural network, we use distributional layers, a neural network pendant to DR proposed by several authors, e.g., Dillon et al. (2017). The idea of distributional layers is simple and compelling. As for DR all or a subset of features are used to estimate each distribution parameter $\theta_k$ of a chosen parametric distribution $D$. Instead of outputting the predicted mean or any other statistic derived from $D$, the outcome $y$ is evaluated on basis of the (probability) density function $p_D$ by calculating the negative log-likelihood $-\log p_D(y|\theta(\nu))$ and using the result as a loss function for backpropagation and updates of weights $w$ and biases $b$.

2.3. Predictor Structure

DR typically assumes the structured predictors $\eta_k$, $k = 1, \ldots, K$ to be an additive decomposition of a linear part $f_0(x) = b + x^T w$ and non-linear functions $f_j(z_j)$. Our SDDL approach additionally allows for one or more deep neural network predictors $d_1(\ldots), \ldots, d_q(\ldots)$ with inputs $u_1, \ldots, u_q$, which can be (partially) shared by the networks and can be (partially) identical to the features $x_1, z_j$:

$$\eta_k = f_0(x) + f_1(z_1) + \ldots + f_r(z_r) + d_1(\ldots) + \ldots + d_q(\ldots).$$

Note that we have suppressed the index for the distributional parameter $\nu$ here to not overload the notation, i.e. to avoid $f_{0,k}, \ldots, d_{q,k}(\ldots)$. The functions $f_j(\cdot)$ represent penalized smooth non-linear effects of features using a basis representation with a function approximation by a linear combination of $L_j$ basis functions. A univariate non-linear effect of feature $z_j$ is, e.g., approximated by $f_j(z_j) \approx \sum_{l=1}^{L_j} B_{j,l}(z_j) w_{j,l}$, where $B_{j,l}(z_j)$ is the $l$th basis function (such as regression splines, polynomial bases or B-splines) evaluated at the observed feature $z_j$. This class is not limited to univariate effects. Tensor product representations allow for two-dimensional non-linear interactions but also non-linear effects of higher-order interactions. Despite being non-linear, the structured predictors are easily understandable due to the additive structure, but often limited to no or only lower order interactions to preserve the interpretability and tractability of the whole model.

2.4. Identifiability

Identifiability is crucial when feature inputs overlap in the structured and unstructured model part. If not constrained
the deep learning effects $d_l$ can also capture linear effects of those features, making the attribution of the effect to either one of the model parts not identifiable. The easiest example is the bias $b$, where if multiple functions $f_1, f_2$ contain different biases $b_1, b_2$, the level of the overall predictor $\eta$ stays the same when subtracting a constant $c$ from $f_1$ and simultaneously adding it to $f_2$. We ensure identifiability of $\eta_k$ by using an orthogonalization operation $C f = 0$, also yielding centered non-linear functions, i.e., $\sum f_j(\cdot) = 0$ to make the model’s overall bias term identifiable (alternative constraints are known to lead to higher uncertainty in subsequent statistical inference (Wood, 2017)). Let $X \in \mathbb{R}^{n \times p}$ be $p$ features with $n$ observations of the structured linear network part and $\mathcal{P}_X \in \mathbb{R}^{n \times n}$ the projection matrix for which $\mathcal{P}_X Z$ is the linear projection of the columns of $Z$ into the space spanned by the linear features for any $Z \in \mathbb{R}^{n \times r}$. To ensure the identifiability of the structured linear part with respect to the non-linear parts of the DNN, the projection into the orthogonal complement $\mathcal{P}_X^\perp = I_n - \mathcal{P}_X$ is applied to the DNN’s outputs $\hat{Z}$ in its second last layer. This yields $\tilde{Z} = \mathcal{P}_X^\perp \hat{Z}$. The result is then multiplied by the weights $\gamma$ of the DNN’s output layer and added to the structured linear network output, yielding the final predictor $X \beta + \tilde{Z} \gamma$ that is fed into the distributional layer. If both, structured and unstructured non-linear parts are present, we first use this orthogonalization operation to ensure identifiability between the linear and non-linear structured parts, combine them and then apply the same operation for separation of the whole structured from the unstructured deep learning predictor.

A schematic architecture of such an operation is depicted as orthogonalization cell in Figure 1, showing one of several possible architectures for combining two structured and one unstructured predictor. While the orthogonalization operation per default attributes as much effect to the structured model part as possible, a tuning parameter $\varpi \in [0, 1]$ can be introduced to control the amount of attribution to the predictors, $\varpi X \beta + (1 - \varpi) \tilde{Z} \gamma$, where $\varpi = 1$ is the default and values smaller 1 attribute more of the linear effect to the non-linear unstructured effect.

3. Inferential Aspects

Having defined our network structure, we now turn to inferential aspects including penalization of structured non-linear terms and corresponding prior assumptions.

3.1. Penalization and Priors

It is believed that neural networks hold an implicit regularization behaviour due to gradient-based optimization (see, e.g., Arora et al., 2019), which we were able to confirm in some of our simulation studies. Despite showing reasonably good non-linear effect estimation in structured additive parts of our network without additional penalization in some cases, we also observed rather coarse estimated non-linear effects or even convergence difficulties when not penalizing structured non-linear effects. We therefore allow for additional penalization of smooth structured effects using kernel regularization in the corresponding layers with tunable smoothing parameter $\lambda_j$ for each smooth term $j \in \{1, \ldots, r\}$ and appropriate penalty matrices $S_j$.

Tuning a large number of smooth terms, potentially in combination with deep learning model parts, is a challenging task. We foster easy tuning by defining the smoothness of each effect in terms of the degrees of freedom (df; see, e.g., Buja et al., 1989) and implement an efficient calculation using the Demmler-Reinsch Orthogonalization (DRO, cf. Ruppert et al., 2003). The latter can easily be parallelized or sped up using a randomized SVD matrix decomposition (Erichson et al., 2019). We thereby also allow for meaningful default penalization by setting $df_j$ to the same value $df^*$ for each effect $j$ and ensure enough flexibility by choosing $df^*$ as the largest value possible among all smoothing terms $df^* = \min_{j \in J} \max df_j$. Note that the maximum for each $df_j$ is given by the corresponding number of columns of the given basis representation. Alternatively in a Bayesian network version of our framework, smoothing can be enforced by prior distribution assumptions on the weights of the basis function. We take advantage of the correspondence between Bayesian model estimation and smoothing splines by using $S_j$ as precision matrix for a zero mean normal prior. Depending on the specification of $f_j$, it is not uncommon that $S_j$ is rank deficient. To guarantee proper posteriors in such cases we add proper priors for the null space of $S_j$.

3.2. Tuning and Convergence

While optimization of the model can be done by minimizing the corresponding negative log-likelihood of the specified distribution, deep distributional regression models can easily under- or overfit the given data. In general, we recommend splitting the data and monitoring the validation loss to ensure generalization ability of the model and, if necessary, to use early stopping. To restrict flexibility in structured predictors, the $df^*$ for each structured non-linear term can be additionally reduced or linear effects can be estimated using $L_1$- or $L_2$-penalties. Moving features from the DNN to one of the structured model parts can further regularize and stabilise the model. As the convergence of weights in the structured layers may be slow, we suggest monitoring both the test error and weights over all epochs to check for either overfitting or non-convergence.

3.3. Bayesian Posterior Estimation

Uncertainty quantification of estimated model weights in DR, also known as epistemic uncertainty, can be accounted
for through Bayesian inference paradigms. As mentioned in Section 3.1 this can be achieved by placing appropriate prior distributions on the model weights. However, resulting posterior distributions are complex and not available in closed form. Different approaches, such as Markov Chain Monte Carlo (MCMC), integrated nested Laplace approximation for a restricted class of models (INLA; Rue et al., 2009) or variational inference (VI; Blei et al., 2017) have been popularized in the literature. The latter has the main advantage to scale well in high-dimensions. We will briefly introduce the concept of VI as a basis for estimation of the proposed network structure in Section 2.

The basic idea of VI is to approximate the augmented posterior \( p(\theta | y) \propto p(y|\theta)p(\theta) \equiv h(\theta) \) with variational density \( q_{\chi}(\theta) \). Here, \( \theta \) denotes a collection of all trainable weights and biases, and \( \chi \) is a vector of variational parameters which are chosen to minimize the Kullback-Leibler divergence between \( q_{\chi} \) and \( p(\theta | y) \). As shown in Ormerod & Wand (2010) this is equivalent to maximizing the variational lower bound \( \mathcal{L}(\chi) = \mathbb{E}_{q}(\log(h(\theta)) - \log(q_{\chi}(\theta))) \) with respect to \( \chi \). VI can be incorporated in our framework using Bayes by Backprop (Blundell et al., 2015), which allows for different combinations of prior for layer weights and variational posterior families with easy interchangeability.

In order to define a distribution over layer weights in the network, existing weight initialization arguments are replaced by distributions over the weights that may, in turn, carry trainable parameters. These so called Bayesian layers (BL; Tran et al., 2019) constitute commonly used layers, such as convolutional layers, and replace their deterministic counterparts with freely selectable positions in the network. Per default, our framework uses these layers to change from a purely frequentist to a Bayesian approach by defining all layers as BL fed into the distributional layer. In other words, by defining priors for the linear predictor of each parameter.

### 3.4. Model Evaluation and Selection

When the goal is to properly predict a distribution of interest, results can be compared on the basis of proper scoring rules (Klein et al., 2015), such as the log-score. These also allow for an objective comparison and selection among different distributions. An equally important aspect in model selection is the choice of features to include in the structured terms of the model’s parameters. In contrast to other regression frameworks, where in particular for high-dimensional settings with more features than observations no one-size-fits-all approach is available, a property of neural networks is their capability of learning more effects than data points available. Combined with the orthogonalization from Section 2.4, our framework thus seamlessly allows for model selection when including all features in all types of predictor parts (i.e., \( x_1, \ldots, x_p \) being equal to \( z_1, \ldots, z_r \) and \( u_1, \ldots, u_q \) with \( p \equiv r \equiv q \)). This can be seen as a variance decomposition approach, where the model separates the linear and non-linear effects (with no interaction) from the remaining deep unstructured effects.

### 4. Experiments on Synthetic Data

We conduct three different simulation specifications with synthetic data to assess the performance of our proposed framework with respect to: a) the goodness-of-fit in terms of the estimation of regression coefficients and structured functional relationships to demonstrate the framework’s capability to properly estimate structured models terms and b) the prediction performance, which will be measured by the predictive log-scores evaluated on the test data set. The first experiment demonstrates the orthogonalization property of the proposed framework, whereas the second simulation compares the estimation results from semi-structured deep distributional learning to various software packages with a particular focus on additive distributional regression. A third experiment investigates estimation performance and epistemic uncertainty quantification of SDDL using different network predictors and mean field VI via BL.

#### 4.1. Orthogonalization and Variance Decomposition

Here we mimic a situation where the practitioner’s interest explicitly lies in decomposing certain feature effects into structured linear, structured non-linear and unstructured non-linear parts, for instance for reasons of interpretability. We simulate 20 data sets with \( n = 1500 \) observations and \( p = 10 \) features \( x_1, \ldots, x_{10} \) drawn from a uniform distribution \( U(-1,1) \). For the outcome \( y \) we consider the cases \( y \sim N(\eta, \sigma^2) \), \( s = 0.1, 1, 10 \) (Normal), \( y \sim Ber(\text{sigmoid}(\eta)) \) (Bernoulli) and \( y \sim Po(\exp(\eta)) \) (Poisson). The predictor \( \eta \) contains each feature as linear effect, non-linear effect and an interaction with the other 9 features. Our framework explicitly models the true linear and non-linear term by separating both effects via orthogonalization and uses a fully connected DNN with two hidden-layers with 32 and 16 hidden units and ReLu activation to account for the interaction. By projecting the output of the second last layer into the orthogonal complement of the structured predictors, we ensure identifiability of the linear and non-linear effects. We do not tune the model but use the DRO approach described in Section 3.1 and a fixed number of epochs.

**Results:** Figure 2 visualizes the estimated and true non-linear relationships between selected features and the outcome. The root mean squared error (RMSE) for the linear effects over different effect sizes ranging from 0.2 to 2 are 0.0145, 0.1725 and 1.2734 for the normal distributions, 0.0394 for the Poisson distribution and 0.6904 for the Bernoulli distribution, which are reasonable small relative
to the effect sizes. Overall the resulting estimates in Figure 2 demonstrate the capability of the framework to recover the partial non-linear effects and only the simulation using a normal distribution with $\text{Var}(y) = 100$, which amounts to a maximum of 4% signal-to-noise ratio (predictor variance divided by noise variance), shows an overfitting behaviour.

4.2. Model Comparison

In this simulation we compare the estimation and prediction performance of our framework with likelihood-based optimization (gamlss), a Bayesian optimization (bamlls) and a model-based boosting (MBB) routine. The different frameworks for distributional regression are available in the software packages gamlss (Rigby & Stasinopoulos, 2005), gamboostLSS (Mayer et al., 2012) and bamlls (Umlauf et al., 2018). For three different distributions (normal, gamma, logistic) we investigate a combination of different number of observations $n \in \{300, 2500\}$ and different number of linear feature effects in the location ($p \in \{10, 75\}$) and 2 linear effects in the scale parameter. In addition to the linear feature effects we add 10 non-linear effects for the location and 2 non-linear effects for the scale.

Results: Table 1 summarizes the mean log-scores of 0.25$n$ test data points and mean RMSE measuring the average of the deviations between true and estimated structured effects. We observe that our approach (SDDL) often yields the best or second to best estimation and prediction performance, while being robust to more extreme scenarios in which the number of observations is small in comparison to the number of feature effects. In this situation other approaches tend to suffer from convergence problems.

4.3. Epistemic Uncertainty

The final simulation has two main purposes. First we want to compare the estimation performance of our framework when specified with and without BL using the MSE. Second, we investigate 90%-credible intervals using BL for smooth terms to assess the validity of inference statements drawn from the estimated variational posterior distribution. Specifically, we report the point-wise coverage rates and their power measured by the point-wise coverage at zero. We use a homoscedastic normal distribution with $n = 5000$ observations and either include 1, 3 or 5 structured non-linear predictors $f_1, \ldots, f_5$ that are penalized using $df^* = 4$. We also investigate the model fits when using an additional 1 hidden layer neural network with 16 hidden units and/or using a Bayesian version of the network.

Results: Results suggest that neither the number of non-linear functions nor the inclusion of a DNN has a notable influence on the performance. By varying the degrees-of-freedom of the smooths in further experiments, we found that an appropriate penalization is essential for both proper estimation and the properties of resulting posterior credible intervals. Table 2 compares the mean squared deviation between estimated and true curves as well as the coverage and power of corresponding intervals. Simulations with variational layers show slightly better estimation performance for the posterior mean curves in comparison to the non-Bayesian smooth estimation. Intervals are too conservative for the first two curves but lack coverage for other smooth functions. Power investigations yield diverse results, but as for coverage, values show better performance for curves for which $df^* = 4$ induces an appropriate smoothing penalty.

Table 1: Median and median absolute deviation of the mean negative predictive log-scores and mean RMSE values of estimated weights and non-linear point estimates across all settings. The best performing approach with smallest measure each is highlighted in bold.

| $n$ | $p$ | bamlss | SDDL | gamlss | MBB |
|-----|-----|--------|------|--------|-----|
| 10  | 1.51 (0.68) | 1.85 (0.55) | 1.02 (0.07) | 4.20 (0.47) |
| 75  | > 10$^{20}$ | 2.84 (0.94) | > 10$^{20}$ | 10.95 |
| 10  | 0.55 (0.06) | 0.96 (0.18) | 0.57 (0.08) | 3.71 (0.24) |
| 75  | 0.64 (0.06) | 1.11 (0.12) | 0.69 (0.07) | 8.85 (0.51) |
| 10  | 1.15 (0.10) | 1.32 (0.31) | 1.04 (0.09) | 1.13 (0.11) |
| 75  | 1.50 (0.36) | 2.34 (0.87) | 2.05 (0.99) | 1.56 (0.15) |
| 10  | 1.01 (0.02) | 0.93 (0.03) | 0.83 (0.02) | 0.96 (0.03) |
| 75  | 1.01 (0.03) | 1.24 (0.05) | 0.84 (0.03) | 1.01 (0.04) |
| 10  | 1.44 (0.14) | 1.75 (0.12) | 4.38 (2.84) | 3.28 (0.37) |
| 75  | 2.55 (0.48) | 2.22 (0.18) | 1.24 (104) | 4.89 (0.26) |
| 10  | 1.7 (0.04) | 1.15 (0.06) | 1.15 (0.04) | > 10$^{20}$ |
| 75  | 2.47 (0.36) | 1.16 (0.08) | 1.23 (0.05) | 4.71 (0.1) |

Table 2: Estimation performance using conventional layers (without BL) or variational layers (with BL) and coverage as well as power results for estimated 90%-credible intervals using BL.

| $f_j$ | MSE |
|-------|-----|
|       | without BL | with BL | Coverage | Power |
| 1     | 0.01 (0.00) | 0.01 (0.00) | 1.00 (0.00) | 0.83 (0.03) |
| 2     | 0.03 (0.00) | 0.03 (0.00) | 0.95 (0.03) | 0.07 (0.19) |
| 3     | 0.09 (0.00) | 0.05 (0.01) | 0.64 (0.06) | 0.12 (0.09) |
| 4     | 0.08 (0.00) | 0.05 (0.01) | 0.70 (0.09) | 0.91 (0.01) |
| 5     | 0.09 (0.00) | 0.09 (0.00) | 0.53 (0.03) | 0.00 (0.00) |
5. Benchmarks and Real-World Data Sets

In addition to the experiments on synthetic data, we provide a number of benchmarks in several real-world data sets in this section. If not stated otherwise, we report measures as averages and standard deviations (in brackets) over 20 random network initializations. Moreover, the hidden and output layers of the DNNs in the SDDL approach use ReLu and linear activation functions per default, respectively.

5.1. Deep Generalized Mixed Models

Tran et al. (2019) use a panel data set with 595 individuals and 4165 observations from Cornwell & Rupert (1988) as an example for fitting deep mixed models by accounting for within subject correlation. Performance is measured in terms of within subject predictions of the log of wage for future time points. We follow their analysis by training the model on the years \( t = 1, \ldots, 5 \) and predicting the years \( t = 6, 7 \). We use a normal distribution with constant variance and model the mean with the same neural network predictor as done by Tran et al. (2019), but exclude the subject ID, which we use as a structured random effect \( \beta_i \) for each individual \( i \): log-wage, \( \log \text{-wage}_{i,t} \sim N(\beta_i + f_1(x_i), f_2(x_i)) \) with \( x_i \) being 12 features also used in Tran et al. (2019), independent and identically distributed random effects \( b_i \sim N(0, \tau^2) \) and \( f_1 \) as well as \( f_2 \) two different DNNs.

Results: Our approach yields an average MSE of 0.04 (0.005) which makes our method competitive to the approach of Tran et al. (2019, MSE=0.05).

5.2. Deep Distributional Models

The first application illustrates the distributional aspect of our framework. Following Rodrigues & Pereira (2018), we consider the motorcycle dataset from Silverman (1985). In contrast to Rodrigues & Pereira (2018), who present a framework to jointly predicting quantiles, our approach models the entire distribution, including a prediction for all quantiles in \((0, 1)\) in one single model.

Results: As we model the distribution itself and not the quantiles explicitly, our approach does not suffer from quantile crossings. Using the quantiles 0.1, 0.4, 0.6 and 0.9, the approach by Rodrigues & Pereira (2018) yields an RMSE of 0.526 (0.003) with an average of 3.050 (0.999) quantile crossings on all test data points. Our approach with DNN for the mean and linear time effect for the distribution’s scale in contrast does not suffer from quantile crossings and yields an RMSE of 0.536 (0.016). A visualization of the fitted mean with selected quantiles is depicted in Figure 3.

5.3. High-Dimensional Classification

Ong et al. (2018) aim at predicting various forms of cancer in high-dimensional gene expression data sets (Colon, Leukaemia and Breast cancer). The authors conduct a Bayesian approach with horseshoe priors and VI, named VAF in the following. We instead use the SDDL approach and combine a linear model with a small DNN (one or two hidden layers and up to 16 hidden units) where we use the proposed orthogonalization operation for the DNN and apply this model to all three data sets with training sample
As an additional comparison, we fit a standard DNN with ordinary least squares and standard neural networks in Song et al. (2019) to benchmark our SDDL against the two 
post-hoc calibration methods isotonic regression (IR; Kuleshov et al., 2018) and the GP-Beta model (GPL; Song et al., 2019) with 16 inducing points. The uncalibrated model for the latter two is a Gaussian process regression (GPR) which performed better than ordinary least squares and standard neural networks in Song et al. (2019). The SDDL assumes a $N(\mu, \sigma^2)$ distribution for the output with $\mu$ and $\sigma^2$. Note that further fine-tuning could be done for SDDL in terms of the output distribution (e.g. house prices are positive and may be skewed rather than symmetric). Here, we only tune the specific structure for the predictors $\eta_\mu = \mu$, $\eta_\sigma = \log(\sigma)$, which consist of structured linear and non-linear effects as well as a DNN for all features, tanh activation functions in the hidden layer(s) and different number of hidden units. We denote those DNNs by $d_\mu(\cdot)$ and $d_\sigma(\cdot)$ with the number of hidden units in brackets. Specifically for the four data sets we define $\eta_\mu$, $\eta_\sigma$ as $b_\mu + f(z_3) + d_\mu(4)$, $\exp(b_\sigma + d_\sigma(4))$ (Diabetes); $b_\mu + x^\top w + \sum_{j=1}^{11} f_j(z_j) + d_\mu(16, 4)$, $\exp(b_\sigma + d_\sigma(2))$, where $x = (z_1, \ldots, z_J)^\top$ (Boston); $b_\mu + \sum_{j \in J} f_j(z_j) + d_\mu(16, 4)$, $\exp(b_\sigma + \sum_{j} \beta_j)$ with index $j \in \{1, 2, 5\}$ being the indices for three of five available numerical features (Airfoil); and $b_\mu + d_\mu(16, 4)$, $\exp(b_\sigma + \sum_{j} \beta_j)$ with index $j \in \{1, \ldots, 12\}$ for each month (Forest Fire). We split the data into 75% for training and 25% for model evaluation, measured by negative log-scores.

**Results:** Table 3 compares the average (standard deviation) of the AUROC on three cancer data sets (first row: Colon cancer; second row: Leukaemia; third row: Breast cancer) for our method, a simple DNN and the VAFC (with different number of factors)

|           | SDDL | DNN | VAFC (4) | VAFC (20) |
|-----------|------|-----|----------|-----------|
| Diabetes  | 1.00 (0.00) | 1.00 (0.00) | 1.00 (0.00) | 1.00 (0.00) |
| Boston    | 0.98 (0.02) | 0.82 (0.23) | 0.91 (0.06) | 0.90 (0.07) |
| Airfoil   | 1.00 (0.00) | 1.00 (0.00) | 0.95 (0.10) | 0.84 (0.12) |

sizes of 42, 38 and 38 and test set sizes of 20, 34 and 4, respectively. The number of features is $p=2000$ for the Colon data, and $p=7129$ for the Leukaemia and Breast datasets. As an additional comparison, we fit a standard DNN with the same architecture as for the SDDL approach but no additional structured predictors, a sigmoid activation function and binary crossentropy loss function.

**Results:** Table 3 compares the average (standard deviation) of the Area under the Receiver Operator Characteristic Curve (AUROC). While all approaches yield an AUROC of one, our SDDL approach is able to outperform the VAFC and standard DNN approach on the other two data set.

### 5.4. Deep Calibrated Regression

Finally, we use four regression datasets (Diabetes, Boston, Airfoil, Forest Fire) analyzed by Song et al. (2019) to benchmark our SDDL against the two post-hoc calibration methods isotonic regression (IR; Kuleshov et al., 2018) and the GP-Beta model (GPL; Song et al., 2019) with 16 inducing points. The uncalibrated model for the latter two is a Gaussian process regression (GPR) which performed better than ordinary least squares and standard neural networks in Song et al. (2019). The SDDL assumes a $N(\mu, \sigma^2)$ distribution
References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16), pp. 265–283, 2016.

Arora, S., Cohen, N., Hu, W., and Luo, Y. Implicit regularization in deep matrix factorization. In Advances in Neural Information Processing Systems, pp. 7411–7422, 2019.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. Variational inference: A review for statisticians. Journal of the American Statistical Association, 112(518):859–877, 2017.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural networks. Proceedings of the 32nd International Conference on Machine Learning, 37:1613–1622, 2015.

Buchholz, A., Wenzel, F., and Mandt, S. Quasi-Monte Carlo variational inference. In Proceedings of the 35th International Conference on Machine Learning, volume 80, pp. 668–677, 2018.

Buja, A., Hastie, T., and Tibshirani, R. Linear smoothers and additive models. The Annals of Statistics, 17(2):453–510, 1989. doi: 10.1214/aos/1176347115.

Cheng, H.-T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Isip, M., et al. Wide & deep learning for recommender systems. In Proceedings of the 1st workshop on deep learning for recommender systems, pp. 7–10. ACM, 2016.

Cornwell, C. and Rupert, P. Efficient estimation with panel data: An empirical comparison of instrumental variables estimators. Journal of Applied Econometrics, 3(2):149–155, 1988.

Dillon, J. V., Langmore, I., Tran, D., Brevdo, E., Vasudevan, S., Moore, D., Patton, B., Alemi, A., Hoffman, M., and Saurous, R. A. Tensorflow distributions. arXiv preprint arXiv:1711.10604, 2017.

Doshi-Velez, F. and Kim, B. Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608, 2017.

Erichson, N. B., Voronin, S., Brunton, S. L., and Kutz, J. N. Randomized matrix decompositions using R. Journal of Statistical Software, 89(11), 2019.

Klein, N., Kneib, T., Lang, S., and Sohn, A. Bayesian structured additive distributional regression with an application to regional income inequality in Germany. Annals of Applied Statistics, 9(2):1024–1052, 2015.

Kuleshov, V., Fenner, N., and Ermon, S. Accurate uncertainties for deep learning using calibrated regression. In Proceedings of the 35th International Conference on Machine Learning, volume 80, pp. 2796–2804, 2018.

Mayr, A., Fenske, N., Hofner, B., Kneib, T., and Schmid, M. Generalized additive models for location, scale and shape for high-dimensional data - a flexible approach based on boosting. Journal of the Royal Statistical Society, Series C - Applied Statistics, 61(3):403–427, 2012.

Ngiam, J., Khosla, A., Kim, M., Nam, J., Lee, H., and Ng, A. Y. Multimodal deep learning. In Proceedings of the 28th International Conference on Machine Learning (ICML), pp. 689–696, 2011.

Ong, V. M.-H., Nott, D. J., and Smith, M. S. Gaussian variational approximation with a factor covariance structure. Journal of Computational and Graphical Statistics, 27(3):465–478, 2018.

Ormerod, J. T. and Wand, M. P. Explaining variational approximations. The American Statistician, 64(2):140–153, 2010.

Rigby, R. A. and Stasinopoulos, D. M. Generalized additive models for location, scale and shape. Journal of the Royal Statistical Society: Series C (Applied Statistics), 54(3):507–554, 2005.

Rodrigues, F. and Pereira, F. C. Beyond expectation: Deep joint mean and quantile regression for spatio-temporal problems. arXiv preprint arXiv:1808.08798, 2018.

Rue, H., Martino, S., and Chopin, N. Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 71(2):319–392, 2009.

Ruppert, D., Wand, M. P., and Carroll, R. J. Semiparametric regression. Cambridge University Press, Cambridge and New York, 2003.

Silverman, B. W. Some aspects of the spline smoothing approach to nonparametric regression curve fitting. Journal of the Royal Statistical Society: Series B (Methodological), 47(1):1–21, 1985.

Song, H., Diethe, T., Kull, M., and Flach, P. Distribution calibration for regression. In Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research, pp. 5897–5906. PMLR, 2019.

Tran, D., Mike, D., van der Wilk, M., and Hafner, D. Bayesian layers: A module for neural network uncertainty. 33rd Conference on Neural Information Processing System (NeurIPS), 2019.
Tran, M.-N., Nguyen, N., Nott, D., and Kohn, R. Bayesian deep net GLM and GLMM. *Journal of Computational and Graphical Statistics*, 2019.

Umlauf, N., Klein, N., and Zeileis, A. Bamlss: Bayesian additive models for location, scale, and shape (and beyond). *Journal of Computational and Graphical Statistics*, 27(3): 612–627, 2018. doi: 10.1080/10618600.2017.1407325.

Wood, S. N. *Generalized additive models: an introduction with R*. Chapman and Hall/CRC, 2017.

Yao, J., Pan, W., Ghosh, S., and Doshi-Velez, F. Weight uncertainty in neural networks. *ICML 2019 Workshop on Uncertainty and Robustness in Deep Learning*, 2019.