Probabilistic Forecasting with Spline Quantile Function RNNs

Jan Gasthaus **Konstantinos Benidis** **Yuyang Wang** **Syama S. Rangapuram**
David Salinas **Valentin Flunkert** **Tim Januschowski**
AWS AI Labs

Abstract

In this paper, we propose a flexible method for probabilistic modeling with conditional quantile functions using monotonic regression splines. The shape of the spline is parameterized by a neural network whose parameters are learned by minimizing the continuous ranked probability score. Within this framework, we propose a method for probabilistic time series forecasting, which combines the modeling capacity of recurrent neural networks with the flexibility of a spline-based representation of the output distribution. Unlike methods based on parametric probability density functions and maximum likelihood estimation, the proposed method can flexibly adapt to different output distributions without manual intervention. We empirically demonstrate the effectiveness of the approach on synthetic and real-world data sets.

1 INTRODUCTION

The problem of forecasting the future values of a time series arises in numerous scientific fields and commercial applications. For example, in medicine, we may wish to predict future glucose levels in patients' blood (Mhaskar et al., 2017), in finance we may be interested in forecasting movement of indices (Engle, 1982) and in retail an accurate forecast of product demand could result in significant cost reductions through optimal inventory management and allocation (Simchi-Levi et al., 1999).

Recently, classical forecasting techniques, such as ARIMA models (Box et al., 2015), or exponential smoothing and its state-space formulation (Hyndman et al., 2008) have been complemented increasingly by deep learning-based techniques that can learn complex patterns across time series and make use of rich metadata without significant manual feature engineering effort (Flunkert et al., 2017; Mukherjee et al., 2018; Qin et al., 2017; Smyl et al., 2018; Bandara et al., 2017; Wen et al., 2017).

Further, probabilistic forecasting, i.e., estimating the entire probability distribution of a time series' future values conditioned on its past instead of just producing a point estimate, is becoming increasingly important in practice, as it allows for the automation of optimal decision making under uncertainty to which forecasting is just a means to an end.

One of the difficulties of applying classical probabilistic forecasting methods as well as recent neural network-based techniques to large scale, real world data sets is the tension between the parametric assumptions made on the output distribution (or the forecast error distribution) and the observed data. The vast majority of techniques assumes a Gaussian distribution—a choice often based on mathematical convenience rather than evidence—which is often not adequate.

In this paper we propose SQF-RNN, a methodology that combines the forecasting capacity of recurrent neural networks (RNNs) with the flexibility of a quantile function-based specification of the observation distribution, designed to overcome this difficulty. SQF-RNN builds on previous advances in RNNs for sequence modeling in general (Graves, 2013; Sutskever et al., 2014), and RNNs for probabilistic forecasting in particular (Flunkert et al., 2017; Wen et al., 2017). Whereas the RNN architecture we employ is similar to that proposed in previous work (Flunkert et al., 2017; Mukherjee et al., 2018; Wen et al., 2017), the novelty lies in the way the output distribution is specified. While previous work either assumes a parametric form for the density with model parameters learned by maximum likelihood estimation or focuses on modeling only a fixed set of quantiles, we employ a spline-based representation of the entire conditional quantile func-

Proceedings of the 22[nd] International Conference on Artificial Intelligence and Statistics (AISTATS) 2019, Naha, Okinawa, Japan. PMLR: Volume 89.

tion and learn parameters by minimizing an appropriate loss function, the continuous ranked probability score (CRPS) (Matheson and Winkler, 1976; Gneiting and Raftery, 2007).

The major contributions of this paper are:

- estimating conditional quantile functions by modeling them using regression splines and learning model parameters by minimizing the CRPS,
- proposing an RNN-based modeling framework for probabilistic forecasting, which makes use of a this spline-based specification of the quantile function of the observation distribution,
- providing a training procedure for the network based on minimizing the CRPS and show how the integral required in computing the CRPS loss for a spline-based quantile function representation can be computed effectively, and,
- demonstrating on several real-world datasets that learning a non-parametric quantile function can be an efficient and robust approach that avoids having to specify a parametric form of the observation distribution a priori.

In Section 2 we review the building blocks of our methodology: quantile functions, the quantile loss, and the CRPS. In Section 3 we describe how these building blocks combine to yield the backbone of our methodology: a spline-based quantile function representation and its estimation by minimizing the CRPS. In Section 4 we describe how this specification of the output distribution is combined with an RNN-based model tailored to the probabilistic forecasting use case. We describe the network architecture as well as the training procedure of the proposed forecasting method. In Section 6 we empirically demonstrate the performance of our method on various datasets. In Section 7 we discuss possible extensions, and conclude the paper in Section 8.

2 PRELIMINARIES

For a real-valued random variable Z, denote by $f_Z(z)$ its probability density function (PDF), and by $F_Z(z) = \int_{-\infty}^{z} f_Z(\zeta)\mathrm{d}\zeta$ its cumulative distribution function (CDF). The associated *quantile function* of Z is defined as

$$F_Z^{-1}(\alpha) = \inf\{z \in \mathbb{R} : \alpha \leq F_Z(z)\}. \tag{1}$$

As indicated by the notation F_Z^{-1}, the quantile function is the inverse of the CDF F_Z if the inverse exists (namely if F_Z is continuous and strictly monotonically increasing). Intuitively, the quantile function $F_Z^{-1} : [0,1] \to \mathbb{R}$ maps a *quantile level* $\alpha \in [0,1]$ to the point z such that the probability that Z takes on values less than z is α. Conversely, for any function $G : [0,1] \to \mathbb{R}$ that is left-continuous and monotonically non-decreasing, there is a unique probability distribution F whose quantile function is $G = F^{-1}$. We denote the set of all such functions by $\mathcal{Q}$.

The quantile function has several appealing properties: Foremost, it can be used directly for reading off prediction intervals and to generate samples from the distribution (by sampling $u \sim \mathrm{Uniform}(0,1)$ and computing $z = F_Z^{-1}(u)$ one obtains a sample z from the distribution F_Z). Further, the monotonicity requirement for quantile functions can be easier to enforce and verify than the constraints imposed e.g. on the PDF, which is required to be non-negative and needs to integrate to one.

2.1 Quantile Regression & Quantile Loss

Given data drawn from a joint distribution $(x, z) \sim F_{(X,Z)}$, one problem with wide applicability is estimating a particular quantile α of the conditional distribution of Z given $X = x$, $r_\alpha(x) = F^{-1}_{Z|X=x}(\alpha)$. Such *quantile regression* problems (Koenker and Bassett Jr, 1978; Koenker, 2005) can be solved by minimizing the *pinball loss* function (also known as *quantile loss*), defined as

$$\Lambda_\alpha(q, z) = (\alpha - \mathcal{I}_{[z<q]})(z - q), \tag{2}$$

where $\mathcal{I}_{[z<q]}$ denotes the indicator function, taking value 1 if $z < q$ and 0 otherwise, $0 < \alpha < 1$ is the quantile level of interest and q the predicted α-th quantile. The key property of this loss function is that the minimizer of its expectation under some distribution F_Z is the α-quantile $F_Z^{-1}(\alpha)$, i.e.,

$$\arg\min_q \mathbb{E}_{z\sim F_Z}[\Lambda_\alpha(q, z)] = F_Z^{-1}(\alpha). \tag{3}$$

In order to estimate $r_\alpha(x)$, we can choose a parameterized family of functions $r_\alpha(x; \theta)$ (e.g. a linear function or neural network with weights θ) as a model for $F^{-1}_{Z|X=x}(\alpha)$ and estimate its parameters θ from the data. By replacing the expectation in (3) with the empirical average, we can estimate the parameters $\theta^\star$ by

$$\theta^\star = \arg\min_\theta \frac{1}{N} \sum_{i=1}^{N} \Lambda_\alpha(r_\alpha(x; \theta), z_i). \tag{4}$$

See Koenker (2005) for more details.

2.2 Continuous Ranked Probability Score (CRPS)

The *continuous ranked probability score* (CRPS) (Matheson and Winkler, 1976; Hersbach, 2000; Gneit-

ing and Raftery, 2007) measures the compatibility of a probability distribution F (represented by its quantile function F^{-1}) with an observation z. It has an intuitive definition[1] as the pinball loss integrated over all quantile levels $\alpha \in [0, 1]$,

$$\mathrm{CRPS}(F^{-1}, z) = \int_0^1 2\Lambda_\alpha(F^{-1}(\alpha), z)\, d\alpha. \quad (5)$$

An important property of the CRPS is that it is a *proper scoring rule* (Gneiting and Raftery, 2007), i.e.,

$$\int g(z)\mathrm{CRPS}(G^{-1}, z)\, dz \leq \int g(z)\mathrm{CRPS}(F^{-1}, z)\, dz \quad (6)$$

for any distributions F and G (with g being the density of G). In other words, if our data is drawn from G, the CRPS will be minimized if the predictive distribution F is equal to G. Further, for deterministic predictions CRPS reduces to the absolute error.

3 SPLINE QUANTILE FUNCTIONS

By minimizing the pinball loss Λ_α we can estimate the conditional quantile $r_\alpha(x)$ for a single level α. If we are interested in multiple levels, say $\alpha_1 < \alpha_2$, we could estimate separate functions r_{α_1} and r_{α_2}, but risk running into the well-known issue of *quantile crossing*, i.e., $\exists x : r_{\alpha_1}(x) > r_{\alpha_2}(x)$. Here, we propose an alternative which does not suffer from this problem, and results in a more expressive description of the entire conditional distribution $F_{Z|X=x}$ and its quantile function $F^{-1}_{Z|X=x}(\alpha)$. To achieve this, we first construct a parameterized family of quantile functions $Q \subset \mathcal{Q}$ indexed by a parameter $\theta \in \Theta \subseteq \mathbb{R}^D$. Given such a family, we can define a family of *conditional quantile functions* $q_\phi(\alpha|x) = q_{\theta(x;\phi)}(\alpha)$ by constructing a mapping from the value of the conditioning variable x to the parameters θ, denoted $\theta(x; \phi)$, which is itself parameterized by $\phi \in \Phi \subseteq \mathbb{R}^{D'}$. While the general framework is applicable to other choices of Q (see e.g (Lang, 2005) for some other options), we focus on linear isotonic regression splines here.

3.1 Linear Isotonic Regression Splines

Linear splines are piecewise-linear functions of the form

$$s(x; \gamma, \mathbf{b}, \mathbf{d}) = \gamma + \sum_{l=0}^{L} b_l (x - d_l)_+, \quad (7)$$

where $\gamma \in \mathbb{R}$ is the intercept term, $\mathbf{b} \in \mathbb{R}^{L+1}$ are weights describing the slopes of the function pieces, $\mathbf{d} \in \mathbb{R}^{L+1}$ is a vector of *knot* positions, i.e., the starting points of the pieces, and $(x)_+ = \max(x, 0)$ is the "hinge" or ReLU function. The number of pieces L is a hyperparameter of the spline.

In order to construct a family of quantile functions from such splines, we re-parameterize the spline function in order to restrict its domain to $[0, 1]$ and to enforce monotonicity.

First, we ensure that the knot positions d_l are ordered, i.e., $d_l < d_{l+1}$, for $l = 0, \ldots, L-1$, and satisfy $0 \leq d_l \leq 1$. We achieve this by setting $d_0 = 0$, and re-parameterizing $d_l = \sum_{l'=1}^{l} \delta_{l'}$, where the $\delta_{l'}$ denote the spaces between the knots, satisfing $\delta_{l'} \geq 0$ and $\sum_{l'=1}^{L} \delta_{l'} = 1$. Next, we need to ensure the monotonicity of the spline. The slope between two knots d_l and d_{l+1} is given by $m_l = \sum_{l'=0}^{l} b_{l'}$. Thus, if we want (7) to be non-decreasing, we need to ensure that $m_l \geq 0, \forall l$. Setting $b_l > 0$ is too restrictive, as this would restrict (7) to convex functions. Instead, we set $b_l = \beta_l - \beta_{l-1}$, where $\beta_l \geq 0$, $l = 0, \ldots, L-1$. One can easily verify that parameterizing b_l in this way yields $m_l = \beta_l \geq 0$ as required.[2] In summary, we have a spline quantile function $q_\theta(\alpha)$ with parameters $\theta = (\gamma, \boldsymbol{\beta}, \boldsymbol{\delta})$, where $\gamma \in \mathbb{R}$, $\delta \in \left\{\mathbf{u} \in [0, 1]^L : \sum_{l=1}^{L} u_l = 1\right\}$ and $\beta \in \mathbb{R}_+^L$ with a total of $2L + 1$ parameters.

3.2 Parameter Estimation by Minimizing CRPS

Analogously to (4) and encouraged by (6), one approach to finding the best-fitting (non-conditional) quantile function $q_{\theta^\star} \in Q$ given data $z_1, \ldots, z_N$ is by minimizing the empirical average of CRPS, i.e.,

$$\theta^\star = \arg\min_{\theta \in \Theta} \frac{1}{N} \sum_{i=1}^{N} \mathrm{CRPS}(q_\theta(\cdot), z_i). \quad (8)$$

This idea can easily be extended to the conditional setting given a mapping $\theta(x, \phi)$ as described above, by minimizing with respect to ϕ instead:

$$\phi^\star = \arg\min_{\phi \in \Phi} \frac{1}{N} \sum_{i=1}^{N} \mathrm{CRPS}(q_\phi(\cdot|x_i), z_i), \quad (9)$$

where $\{(x_i, z_i)\}_{i=1}^N$ is the data given. Note that by construction $q_\phi(\cdot|x)$ is a valid quantile function $q_\phi(\cdot|x) \in \mathcal{Q}$ for any x, so that quantile crossing can not occur.

[1] We define the CRPS here as a loss function in negative orientation, i.e., lower is better. Other equivalent definitions of the CRPS exist, e.g. as the average of Brier scores, $\mathrm{CRPS}(F, z) = \int_{-\infty}^{\infty} (F(q) - \mathcal{I}_{[z<q]})^2 dq$, or $\mathrm{CRPS}(F, z) = E_F|X - z| - E_F \frac{1}{2}|X - X'|$. See (Hersbach, 2000; Gneiting and Raftery, 2007) for more details.

[2] Note d_L and b_L can be omitted when computing $s(\cdot)$, as we always have $d_L = 1$, and b_L does not affect the value of $s(\cdot)$ on $[0, 1]$.

3.3 Evaluating the CRPS Integral

In order to solve the optimization problem (9) e.g. through gradient-based stochastic optimization, we need to evaluate the CRPS integral (5) and its derivatives wrt. F^{-1}. In general, this may require numerical integration, e.g. a simple Monte-Carlo estimate by sampling random quantile levels α uniformly, or more sophisticated quadrature techniques such as Clenshaw-Curtis quadrature (Clenshaw and Curtis, 1960). Fortunately, in some special cases, such as when F^{-1} is given by a piecewise linear spline of the form (7) with monotonicity restrictions imposed as described above, the integral can be evaluated analytically. In particular, for the piecewise-linear spline (7) we have,

$$\int_0^1 2\Lambda_\alpha(s(\alpha;\gamma,\mathbf{b},\mathbf{d}),z)d\alpha = (2\tilde{a}-1)z + (1-2\tilde{a})\gamma + \sum_{l=0}^{L} b_l \left(\frac{1-d_l^3}{3} - d_l - \max(\tilde{a},d_l)^2 + 2\max(\tilde{a},d_l)d_l\right). \tag{10}$$

Here, $\tilde{a}$ is the quantile level such that $s(\tilde{a};\gamma,\mathbf{b},\mathbf{d}) = z$, given by

$$\tilde{a} = \frac{z - \gamma + \sum_{l=0}^{l_0} b_l d_l}{\sum_{l=0}^{l_0} b_l}, \tag{11}$$

where $l_0 = \max\{l | s(d_l;\gamma,\mathbf{b},\mathbf{d}) < z, 0 \le l \le L\}$, with $d_0 = 0$, which can be easily found in $O(L)$ time. See the supplement for more details.

4 FORECASTING WITH SQF-RNN

We now turn to how such a spline-based representation of the quantile function can be combined with recurrent neural networks and applied to probabilistic forecasting, yielding our proposed model SQF-RNN. The general probabilistic forecasting problem is the following. We are given a set $\{z_{1:T_i}\}_{i=1}^N$ of N univariate time series, where $z_{1:T_i} = (z_1, z_2, \ldots, z_{T_i})$ and $z_t \in \mathbb{R}$ denotes the value of the i-th time series at time t. Further, let $\{\mathbf{x}_{1:T_i+\tau}\}_{i=1}^N$ be a set of associated, time-varying covariate vectors with $\mathbf{x}_t \in \mathbb{R}^D$.[3] Our goal is to produce a set of probabilistic forecasts, i.e., for each $i = 1, \ldots, N$ we are interested in (some representation of) the probability distribution of future trajectories $z_{T_i+1:T_i+\tau}$ given the past:

$$p\left(z_{T_i+1:T_i+\tau} \,|z_{1:T_i}, \mathbf{x}_{1:T_i+\tau}; \phi\right). \tag{12}$$

Here ϕ denotes the learnable parameters of the model, which are shared between and learned jointly from all N time series.

[3]Note that covariates are assumed to be available also during prediction. They can, for example, encapsulate information about holidays, product promotions, etc. that is known in advance.

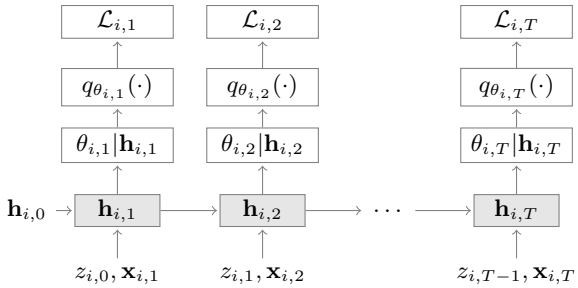

Figure 1: Training: At each time step t, the inputs to the network are the covariates $x_{i,t}$, the target value at the previous time step $z_{i,t-1}$, as well as the previous network output $\mathbf{h}_{i,t-1}$. The network output $\mathbf{h}_{i,t} = r(\mathbf{h}_{i,t-1}, z_{i,t-1}, \mathbf{x}_{i,t})$ is then fed to the projection layer, which outputs the parameters $\theta_{i,t} = \theta(\mathbf{h}_{i,t}, \phi)$ that define the spline function. Finally, the spline function and the target value $z_{i,t}$ are used to compute the CRPS loss and train the model.

At a high level, our modeling setup is the following: The time series $z_{i,t}$ and covariates $\mathbf{x}_{i,t}$ are provided as input to an autoregressive LSTM-based recurrent neural network whose architecture follows that described in (Flunkert et al., 2017). The network is autoregressive and recurrent in the sense that it takes as an input the value of the time series in the previous time step *and* the previous state of the network. The real-valued output of the network is passed through a projection layer that provides a set of parameters which define a spline quantile function. Finally, the CRPS is used as a loss function, measuring the fit of the quantile function to the observed time series data points. In order to produce forecasts, Monte Carlo samples from (12) are generated through sequential sampling from the quantile functions.

The model architecture and the training and inference procedures of the model are illustrated in Figures 1 and 2.

Our objective is to estimate the conditional quantile function

$$F^{-1}\left(z_{i,t}|\mathbf{z}_{i,1:t-1}, \mathbf{x}_{i,1:T_i+\tau}\right) \tag{13}$$

for $t = T_i + 1, \ldots, T_i + \tau$ given all the past observations and all the covariates. For this, we consider a spline quantile function model $q_{\theta(\mathbf{h}_{i,t},\phi)}(\alpha) \in \mathcal{Q}$. The parameters θ are functions of the output $\mathbf{h}_{i,t}$ of an autoregressive recurrent neural network (Graves, 2013), with

$$\mathbf{h}_{i,t} = r(\mathbf{h}_{i,t-1}, z_{i,t-1}, \mathbf{x}_{i,t}, \phi). \tag{14}$$

Here, the function $r(\cdot)$ is a multi-layer recurrent neural network with LSTM cells (as described in (Flunkert et al., 2017)), and ϕ denotes the set of parameters of

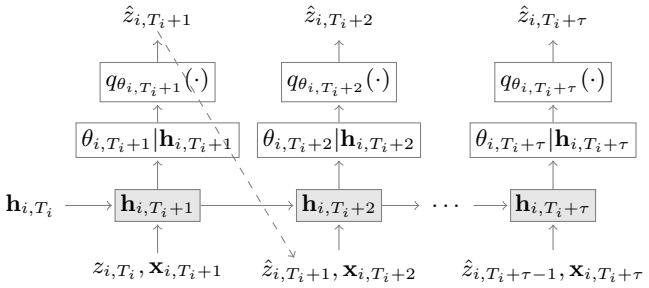

Figure 2: Inference: In the conditioning range ($t \leq T_i$) the (known) history of the time series $z_{i,t}$ is fed in to the network, along with the corresponding covariates. In the prediction range ($t > T_i$) a sample $\hat{z}_{i,t} = q_{\theta(\mathbf{h}_{i,t})}(\alpha)$, with $\alpha \sim \text{Uniform}(0,1)$, is drawn and provided as input for the time step, until the end of the prediction range $t = T + \tau$, generating one sample path. Repeating this prediction process yields samples from the joint prediction distribution.

the neural network. The mapping $\theta(\cdot)$ takes the real-valued network output and maps it to the (possibly restricted) domain of the quantile function parameters. For the spline quantile function described in Section 3, we apply an affine transformation followed by a softmax and softplus transformation to obtain $\boldsymbol{\delta}$ and $\boldsymbol{\beta}$, respectively.

During training, we split each time series into fixed-size windows with different starting points, thereby generating multiple training instances. Each window has a fixed length T, spanning the conditioning and the prediction range. For each window, we then compute the loss $\mathcal{L}_i = \sum_t \mathcal{L}_{i,t}$, where

$$\mathcal{L}_{i,t} = \text{CRPS}(q_{\theta(\mathbf{h}_{i,t},\phi)}(\cdot), z_{i,t}) \tag{15}$$

and $t = 1, \ldots, T$ correspond to the time stamps contained in the chosen window. This loss is then minimized using the Adam (Kingma and Ba, 2014) adaptive stochastic gradient descent optimizer.

During inference, the time series values in the prediction range, i.e., $\mathbf{z}_{i,T_i:T_i+\tau}$, $\forall i$, are unknown and the network output $\mathbf{h}_{i,t}$ cannot be computed for $t \geq T_i$. To obtain Monte Carlo sample paths from the model, at each time step $t \geq T_i$ we generate a single sample $\hat{z}_{i,t} = q_\theta(\mathbf{h}_{i,t}, \phi^\star)(\alpha)$, with $\alpha \sim \text{Uniform}(0,1)$, and use it as input to the network for the subsequent time step. By repeating this process we construct multiple sample paths, which we can use to evaluate any empirical confidence interval or specific quantile value of the distribution (12). Figure 2 illustrates this inference procedure.

5 RELATED WORK

Probabilistic modeling with quantile functions—while not widely used in Machine Learning—has been explored in the literature for a wide range of applications (see e.g. (Gilchrist, 2000) and references therein). Quantile regression, i.e., estimating quantiles of conditional distributions, has also been studied extensively, both in terms of theory as well as applications in numerous disciplines, with a large body of literature summarized in (Koenker, 2005).

One of the first approaches that uses neural networks to estimate quantiles appears in (Taylor, 2000) where a quantile regression neural network (QRNN) is proposed. In QRNN, for a fixed quantile level, the quantile loss with some regularization terms is minimized and the model is applied to estimate quantiles of multi-period returns. Cannon (2011) provide an R implementation. The literature contains further variants (Feng et al., 2010; Xu et al., 2016) all of which suffer from the quantile crossing problem.

Hatalis et al. (2017) propose another approach to estimating multiple quantiles simultaneously using a NN model that minimizes a smooth approximation of the pinball loss function (Zheng, 2011). The network outputs a list of preselected quantiles instead of a single one, however there is no guarantee that there will be no quantile crossover (although a heuristic to reduce this effect is proposed). Xu et al. (2017) propose a NN that outputs the value of a given quantile and is trained using the pinball loss (and the Huber version of it (Cannon, 2011)) for a grid of different quantiles. However, none of these methods are directly applicable to sequential (time series) data.

Flunkert et al. (2017) propose DeepAR, a sequence-to-sequence probabilistic forecasting model with RNNs. DeepAR outputs the parameters of a distribution and is trained with maximum likelihood estimation. Sequential sampling is applied to provide probabilistic forecasts. This method requires the user to specify the density function of an output distribution that fits the data. Wen et al. (2017) propose a multi-horizon quantile RNN (MQ-RNN) forecaster, that does not require user to postulate a distribution. MQ-RNN estimates a number of preselected quantile values while using a multi-horizon strategy, i.e., it models a multivariate target that corresponds to the future values of the time series instead of generating samples and feeding them back to the network in order to estimate the subsequent values. Again, this approach suffers from the quantile crossing problem.

Our approach is closely related to (Flunkert et al., 2017; Mukherjee et al., 2018) in that we that we em-

ploy an almost identical RNN architecture. However, we do no assume a pre-defined standard density function (Flunkert et al., 2017) or a mixture of Gaussians (Mukherjee et al., 2018; Bishop, 1994). Neither do we prescribe specific quantile values (Wen et al., 2017). Instead, we optimize the network based on CRPS which takes into account the whole range of quantiles and we avoid quantile crossing by construction.

While the CRPS is commonly used to evaluate a given trained model, its direct usage as a loss function for learning is rare. Gneiting et al. (2005) provides one of the few exceptions, where the CRPS is minimized to learn the coefficients of an ensemble model. However, a normal distribution is assumed, for which the CRPS also has an analytic form.

More recently, Ostrovski et al. (2018) proposed generative modeling via quantile functions and minimizing the CRPS. While shown to be effective for the considered domain (image modeling), they did not restrict their quantile function approximation to be non-decreasing, thereby also being prone to quantile crossing.

6 EXPERIMENTS

The following experiments with synthetic and real-world data provide evidence for the practical effectiveness of our approach.

6.1 Distribution Recovery

First, we analyse the distribution recovery ability of the proposed SQF-RNNmodel. For this, we generate the value at time t for time series i as follows:

$$z_{i,t} = n_{i,t}, \quad \forall i, t, \tag{16}$$

where $n_{i,t}$ follows a fixed Gaussian Mixture Model, GMM($\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\sigma}$), with $\boldsymbol{\pi} = [0.3, 0.4, 0.3]^\top$, $\boldsymbol{\mu} = [-3, 0, 3]^\top$, and $\boldsymbol{\sigma} = [0.4, 0.4, 0.4]^\top$ being the mixture coefficients, the mean, and the standard deviation of the components, respectively. We create 500 time series with 8736 observations each, which corresponds to one year of hourly observations (i.e., $24 \times 7 \times 52$).

We use the above synthetic dataset to train our model using a conditioning range of 96 and a prediction range of 24. Based on the above generation process, the conditional distribution of each time step is identical to the noise distribution. Figure 3 shows the (true) noise distribution in comparison to the distribution of the generated samples during inference. We observe that the generated samples follow the true distribution closely. This indicates that the model is able to adapt and capture the complex nature of the true distribution. Note that the model has no information about the true distribution apart from the actual time series samples, i.e., there was no explicit modeling of the GMM or the number of components. This shows the robustness of the proposed method because it can capture complex multi-modal densities without any prior information or assumptions.

6.2 Empirical Study

We evaluate the performance of SQF-RNN model on the following datasets:

- `elec`: hourly time series of the electricity consumption of 370 customers (Dheeru and Karra Taniskidou, 2017).
- `traffic`: hourly occupancy rate, between 0 and 1, of 963 car lanes of San Fransisco bay area freeways (Yu et al., 2016).
- `wiki`: daily count time series of number of hits of 9013 wiki pages.
- `dom`: weekly product gross margin in percent of 10^5 products in various stores. Subset of the full dominick's dataset.[4]

Note that `elec` and `traffic` are highly-regular data sets whereas `wiki` and `dom` are less regular and overall more challenging to forecast accurately for. Section B in the supplement contains further details.

To assess the accuracy of SQF-RNN we compare with DeepAR (Flunkert et al., 2017) as implemented in (Januschowski et al., 2018) using the student-T distribution, and the ETS method from (Hyndman et al., 2007).[5]

First, we use metrics that quantify the quantile prediction accuracy of the algorithms. In particular, we compute the mean quantile loss (QLm), the 50-th and 90-th percentile loss (QL50 and QL90, respectively), and the mean scaled interval score (MSIS) for a 95% prediction interval. The exact definitions of all metrics used are provided in the supplementary material. Table 1 summarizes the results over these metrics of all the algorithms and datasets.

The results in Table 1 show that for the regular data sets, SQF-RNN and DeepAR-t only differ marginally.

[4] https://research.chicagobooth.edu/kilts/marketing-databases/dominicks

[5] We also performed preliminary experiments with Prophet (Taylor and Letham, 2017), but concluded that manual intervention was needed to provided competitive results, and hence refrain from reporting results obtained with default settings here.

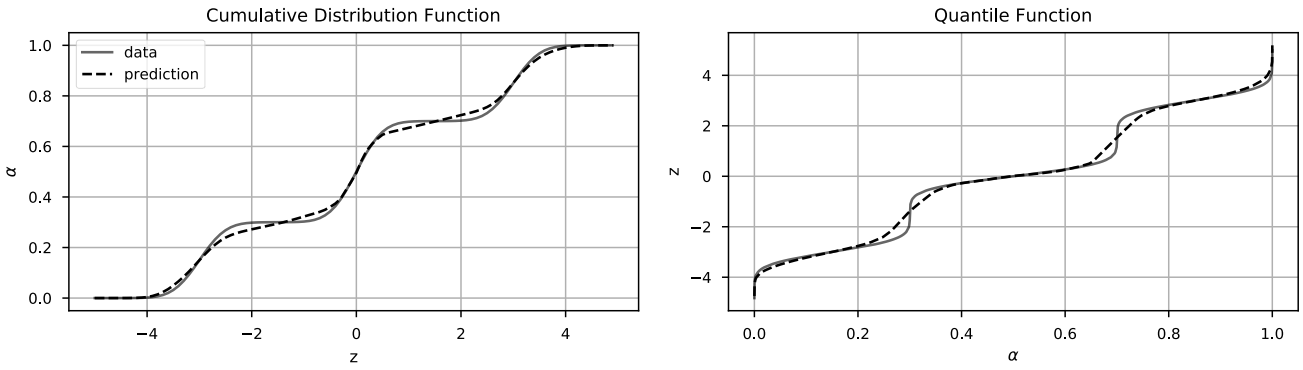

Figure 3: Distribution recovery experiment. Data drawn iid. from a three-component mixture of Gaussians is used to train the model, and prediction sample paths are obtained. The left plot shows the recovered marginal CDF of the samples and of the true data distribution, while the right plot shows the corresponding quantile functions. As can be seen, the model learns parameters of the linear spline which recover this multi-modal distribution.

This is due to the regular nature of the data which fits the modeling assumptions in DeepAR-t. ETS performs well for `elec`, but not for `traffic`. The latter is due to the weekly seasonality, which cannot be modeled by the implementation of ETS used.

We note that for the data sets that are less regular, `wiki` and `dom`, SQF-RNN outperforms the other methods, in particular for the extreme quantile scores and the overall distribution shape. This confirms the experiments in Section 6.1 on real-world data.

Apart form the quantile related metrics, we assess the performance of the algorithms using metrics that quantify the point estimate prediction accuracy. In particular we compute the Normalized Root Mean Square Error (NRMSE), the symmetric Mean Absolute Percentage Error (sMAPE) and the Mean Absolute Scaled Error (MASE). Since the algorithms do not produce directly point forecasts we use the median forecast values for the evaluation of these metrics. Table 1 summarizes the results over these metrics of all the algorithms and datasets. We note that overall, SQF-RNN and DeepAR-t behave roughly similar wrt. the point forecasts accuracy metrics and both outperform ETS consistently. This is additional evidence for the robustness of our method, in the sense that the increased accuracy for probabilistic metrics does not come at the expense of reduced point forecast accuracy.

In a final experiment, we evaluate the proposed SQF-RNN model on the M4 forecasting competition dataset[6], which consists of 10^5 time series. Following the competition guidelines, we compute the Overall Weighted Average (OWA) of the sMAPE and MASE metrics, defined as

$$\text{OWA} = 0.5\frac{\text{sMAPE}}{0.15201} + 0.5\frac{\text{MASE}}{1.685}, \tag{17}$$

which assesses the point-forecast ability of the algorithms, and the MSIS metric for a 95% prediction interval. We compare SQF-RNN with default parameters (given in Appendix B) with the winning model of the competition, Smyl et al. (2018). In Table 2 we summarize the results. We observe that, even without any hyperparameter optimization or model adaptations for the M4 datasets, OWA is less than 8% worse than the winning model, which is finely tuned for the M4 competition, showing the robustness and flexibility of SQF-RNN. Further, the MSIS result of SQF-RNN would have ranked second in the M4 competition, showing again the effectiveness of the model.

7 EXTENSIONS

In order to clarify the exposition, we have focused on a particular class of parameterized quantile functions, namely linear isotonic splines. In preliminary experiments we have explored other classes of spline functions, namely sigmoidal splines and I2-splines, which can be restricted to non-decreasing functions in a similar way (Lang, 2005), and might be more appropriate for a given application. Further, while we have focused on the application to probabilistic forecasting, the approach is more generally applicable to modeling conditional distributions or otherwise related quantile functions. One such application is robust (linear) regression, as a direct extension of quantile regression approaches (Koenker, 2005) to multiple quantiles while avoiding quantile crossing.

In previous work on quantile regression with neural

[6]https://www.m4.unic.ac.cy/

dataset	method	probabilistic metrics				point metrics		
		QLm	QL50	QL90	MSIS	NRMSE	sMAPE	MASE
elec	SQF-RNN	**0.049**	**0.066**	0.035	10.21	**0.518**	0.113	0.937
	DeepAR-t	0.051	0.068	**0.033**	**7.77**	0.55	**0.110**	**0.931**
	ETS	0.076	0.100	0.050	9.992	0.838	0.156	1.247
traffic	SQF-RNN	**0.093**	0.119	**0.090**	**5.25**	**0.381**	0.117	0.449
	DeepAR-t	**0.093**	**0.117**	**0.090**	5.54	0.396	**0.104**	**0.442**
	ETS	0.427	0.488	0.325	20.856	0.872	0.594	1.881
wiki	SQF-RNN	**0.252**	**0.306**	**0.303**	**15.68**	**3.126**	**0.219**	**1.072**
	DeepAR-t	0.275	0.325	0.350	19.35	3.188	0.254	1.136
	ETS	0.788	0.440	0.836	61.685	3.261	0.301	2.214
dom	SQF-RNN	**0.318**	0.438	**0.227**	**9.143**	1.015	1.466	0.847
	DeepAR-t	0.347	**0.426**	0.314	17.57	0.993	1.447	**0.819**
	ETS	0.471	0.485	0.358	16.387	**0.954**	**1.434**	0.967

Table 1: Comparison against competing methods based on various probabilistic as well point metrics.

method	sMAPE	MASE	OWA	MSIS
SQF-RNN	0.1244	1.60	0.885	14.09
Smyl et al. (2018)	0.1137	1.54	0.821	12.23

Table 2: M4 competition results.

networks, a smoothed version of the pinball loss—essentially an asymmetric version of the Huber loss—has been proposed, to avoid the non-differentiability of the pinball loss function at zero (Cannon, 2011). We have experimented with a variant of the CRPS based on this quantile loss function in our framework, but have not found it to be advantageous in practice.

In some settings, the model as described might be too flexible, and one may wish to impose further restrictions on the mapping $\theta(\cdot)$. In particular, in the forecasting setting, a reasonable assumption might be that the individual conditional distributions $p(z_{i,t}|z_{i,1:t-1}, \mathbf{x}_{i,t-1})$ vary in their location and scale, but have a common shape. Such restrictions are easy to incorporate through suitable re-parameterizations of the spline function. In practice, we have found the restriction to a common shape to be effective for smaller data sets.

Finally, in practice one may wish to emphasize certain quantiles over others. This can easily be incorporated by using a quantile-weighted version of CRPS, as proposed by Gneiting and Ranjan (2011).

8 CONCLUSIONS

We have proposed a framework for modeling conditional quantile functions using isotonic splines, and shown how their parameters can be learned by minimizing the CRPS. We have described an RNN-based probabilistic forecasting model based on this idea, and have demonstrated its effectiveness on artificial and real data sets.

We think the results obtained with our approach are promising, and warrant further investigation of models combining conditional quantile function estimation and deep learning.

Modeling the quantile function directly, as opposed to using a flexible parametric density model (e.g. a mixture of Gaussians), can be advantageous, especially when the quantile function is the object of interest or when the resulting optimization problem is more stable or easier to solve. By restricting the quantile function to a particular form (e.g. a linear spline) one imposes a particular inductive bias, and it depends on the application whether this bias is suitable.

References

Bandara, K., Bergmeir, C., and Smyl, S. (2017). Forecasting across time series databases using long short-term memory networks on groups of similar series. *CoRR*, abs/1710.03222.

Bishop, C. M. (1994). Mixture density networks. Technical Report NCRG/4288, Aston University, Birmingham, UK.

Box, G. E. P., Jenkins, G. M., Reinsel, G. C., and Ljung, G. M. (2015). *Time series analysis: forecasting and control*. John Wiley & Sons.

Cannon, A. J. (2011). Quantile regression neural networks: Implementation in R and application to precipitation downscaling. *Computers & geosciences*, 37(9):1277–1284.

Clenshaw, C. W. and Curtis, A. R. (1960). A method

for numerical integration on an automatic computer. *Numerische Mathematik*, 2(1):197–205.

Dheeru, D. and Karra Taniskidou, E. (2017). UCI machine learning repository.

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation. *Econometrica: Journal of the Econometric Society*, page 9871007.

Feng, Y., Li, R., Sudjianto, A., and Zhang, Y. (2010). Robust neural network with applications to credit portfolio data analysis. *Statistics and its interface*, 3(4):437.

Flunkert, V., Salinas, D., and Gasthaus, J. (2017). DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *arXiv preprint arXiv:1704.04110*.

Gilchrist, W. (2000). *Statistical modelling with quantile functions*. Chapman and Hall/CRC.

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

Gneiting, T., Raftery, A. E., Westveld III, A. H., and Goldman, T. (2005). Calibrated probabilistic forecasting using ensemble model output statistics and minimum crps estimation. *Monthly Weather Review*, 133(5):1098–1118.

Gneiting, T. and Ranjan, R. (2011). Comparing density forecasts using threshold-and quantile-weighted scoring rules. *Journal of Business & Economic Statistics*, 29(3):411–422.

Graves, A. (2013). Generating sequences with recurrent neural networks. *arXiv preprint arXiv:1308.0850*.

Hatalis, K., Lamadrid, A. J., Scheinberg, K., and Kishore, S. (2017). Smooth pinball neural network for probabilistic forecasting of wind power. *arXiv preprint arXiv:1710.01720*.

Hersbach, H. (2000). Decomposition of the continuous ranked probability score for ensemble prediction systems. *Weather and Forecasting*, 15(5):559–570.

Hyndman, R., Koehler, A. B., Ord, J. K., and Snyder, R. D. (2008). *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media.

Hyndman, R. J., Khandakar, Y., et al. (2007). *Automatic time series for forecasting: the forecast package for R*. Number 6/07. Monash University, Department of Econometrics and Business Statistics.

Januschowski, T., Arpin, D., Salinas, D., Flunkert, V., Gasthaus, J., Stella, L., and Vazquez, P. (2018). Now available in amazon sagemaker: Deepar algorithm for more accurate time series forecasting. *https://aws.amazon.com/blogs/machine-learning/now-available-in-amazon-sagemaker-deepar-algorithm-for-more-accurate-time-series-forecasting/*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Koenker, R. (2005). *Quantile Regression*. Econometric Society Monographs. Cambridge University Press.

Koenker, R. and Bassett Jr, G. (1978). Regression quantiles. *Econometrica: Journal of the Econometric Society*, pages 33–50.

Lang, B. (2005). Monotonic multi-layer perceptron networks as universal approximators. In *International Conference on Artificial Neural Networks*, pages 31–37. Springer.

Matheson, J. E. and Winkler, R. L. (1976). Scoring rules for continuous probability distributions. *Management science*, 22(10):1087–1096.

Mhaskar, H. N., Pereverzyev, S. V., and van der Walt, M. D. (2017). A deep learning approach to diabetic blood glucose prediction. *Frontiers in Applied Mathematics and Statistics*, 3:14.

Mukherjee, S., Shankar, D., Ghosh, A., Tathawadekar, N., Kompalli, P., Sarawagi, S., and Chaudhury, K. (2018). ARMDN: associative and recurrent mixture density networks for eretail demand forecasting. *CoRR*, abs/1803.03800.

Ostrovski, G., Dabney, W., and Munos, R. (2018). Autoregressive quantile networks for generative modeling. *arXiv preprint arXiv:1806.05575*.

Qin, Y., Song, D., Cheng, H., Cheng, W., Jiang, G., and Cottrell, G. W. (2017). A dual-stage attention-based recurrent neural network for time series prediction. *CoRR*, abs/1704.02971.

Simchi-Levi, D., Simchi-Levi, E., and Kaminsky, P. (1999). *Designing and managing the supply chain: Concepts, strategies, and cases*. McGraw-Hill New York.

Smyl, S., Ranganathan, J., and Pasqua, A. (2018). M4 forecasting competition: Introducing a new hybrid ES-RNN model. `https://eng.uber.com/m4-forecasting-competition/`.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, pages 3104–3112.

Taylor, J. W. (2000). A quantile regression neural network approach to estimating the conditional density of multiperiod returns. *Journal of Forecasting*, 19(4):299–311.

Taylor, S. J. and Letham, B. (2017). Forecasting at scale. *The American Statistician*.

Wen, R., Torkkola, K., and Narayanaswamy, B. (2017). A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053*.

Xu, Q., Deng, K., Jiang, C., Sun, F., and Huang, X. (2017). Composite quantile regression neural network with applications. *Expert Systems with Applications*, 76:129–139.

Xu, Q., Liu, X., Jiang, C., and Yu, K. (2016). Quantile autoregression neural network model with applications to evaluating value at risk. *Applied Soft Computing*, 49:1–12.

Yu, H.-F., Rao, N., and Dhillon, I. S. (2016). Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in neural information processing systems*, pages 847–855.

Zheng, S. (2011). Gradient descent algorithms for quantile regression with smooth approximation. *International Journal of Machine Learning and Cybernetics*, 2(3):191.