Ensemble Conformalized Quantile Regression for Probabilistic Time Series Forecasting

Vilde Jensen, Filippo Maria Bianchi*, Stian Normann Anfinsen

Abstract—This paper presents a novel probabilistic forecasting method called *ensemble conformalized quantile regression (EnCQR)*. EnCQR constructs distribution-free and approximately marginally valid prediction intervals (PIs), which are suitable for nonstationary and heteroscedastic time series data. EnCQR can be applied on top of a generic forecasting model, including deep learning architectures. EnCQR exploits a bootstrap ensemble estimator, which enables the use of conformal predictors for time series by removing the requirement of data exchangeability. The ensemble learners are implemented as generic machine learning algorithms performing quantile regression, which allow the length of the PIs to adapt to local variability in the data. In the experiments, we predict time series characterized by a different amount of heteroscedasticity. The results demonstrate that EnCQR outperforms models based only on quantile regression or conformal prediction, and it provides sharper, more informative, and valid PIs.

Index Terms—Probabilistic forecasting; time series analysis; uncertainty quantification; conformal prediction; quantile regression; heteroscedasticity; ensemble learning; deep neural networks.

I. INTRODUCTION

In real-world planning and decision-making processes, accurate time series forecasting is essential and it is often desirable to express the uncertainty in the predictions through a *probabilistic* forecast [1]. The most common way of obtaining probabilistic forecasts is by generating prediction intervals (PIs), which provide an admissible range of values for future observations with a specified confidence level [2]. PIs are termed *valid* if the coverage probability of future observations matches the specified confidence level. The PI width is governed by the confidence level, where more uncertain predictions produce wider intervals, but also by the performance of the underlying prediction algorithm. When PIs become overly wide, they are less informative and denote poor performance in the prediction model. Thus, probabilistic forecasts should yield PIs that are as narrow as possible, while ensuring the designed confidence level [1].

In application domains such as energy analytics, time series often exhibit strong seasonal and heteroscedastic behavior, as the variance in the observations relate to the cyclic nature of the data [3], [4]. This makes some time intervals harder to predict, e.g., the variation in energy consumption is typically higher and more volatile during the day than in the night. When the variability is lower, one can make more confident predictions with narrower PIs. However, methods that construct fixed-length PIs are not able to model such a variability in the uncertainty and assign the same uncertainty to all time steps, often resulting in overly conservative PIs [5]. To obtain more informative PIs for heteroscedastic time series, the length of the PIs should adapt to the variability at each time step.

Contributions. We directly tackle the challenge of constructing adaptive and valid PIs for time series data by combining and leveraging the strengths of quantile regression (QR) and conformal prediction (CP).

CP is a probabilistic forecasting technique that constructs valid PIs in finite samples without making any distributional assumptions besides observations being exchangeable, which makes it unsuitable for time series data. To apply CP to time series data we rely on the *leave-one-out* ensemble prediction theory [6]. In addition, by using ensemble learners that perform QR, we generate PIs that adapt to the local variability in the time series.

The proposed method, called *ensemble conformalized quantile regression* (EnCQR), is flexible, as it can be placed on top of any QR algorithm. In addition, EnCQR is distribution-free and it constructs approximately valid PIs independently of the accuracy of the underlying prediction method.

	Adaptive PI	Valid PI
QR	✓	✗
CP	✗	✓
EnCQR	✓	✓

We test the performance of EnCQR on five real-world datasets from application domains where it is common to encounter heteroscedastic time series. We show that EnCQR can produce valid and adaptive PIs for such data. To demonstrate the versatility of EnCQR, we use it on top of three different regression algorithms: random forest regression and two neural networks for time series data. Results show that, compared to the state of the art, EnCQR generates narrower yet valid PIs.

II. BACKGROUND

A. Prediction Intervals (PIs)

Let $X \in \mathcal{X}$ and $Y \in \mathcal{Y}$ be random variables representing the input observation and label, respectively. We denote with $\pi(X, Y)$ the joint distribution of X and Y, and with $\pi(Y|X)$ the conditional distribution of Y given X. A PI constructed using a collection of training samples

Correspondence: *filippo.m.bianchi@uit.no

V. Jensen did this work at Department of Physics and Technology, UiT The Arctic University of Norway, and is now with Kongsberg Satellite Services.

F. M. Bianchi is with the Department of Mathematics and Statistics, UiT The Arctic University of Norway and NORCE Norwegian Research Centre.

S. N. Anfinsen is with NORCE Norwegian Research Centre and Department of Physics and Technology, UiT The Arctic University of Norway.

$\{(x_i, y_i), i = 1, 2, \ldots, n\}$, where (x_i, y_i) are realizations of $\pi(X, Y)$, is given by $\hat{C}_{\pi,n}(x) = [L(x), U(x)]$, where L and U are functions that map $\mathcal{X}$ into $\mathcal{Y}$. The width, or length, of a PI $U(X_{n+1}) - L(X_{n+1})$ is governed by the confidence level α, i.e. the probability $(1 - \alpha)$ of a new observation lying within the PI. More uncertain predictions produce wider intervals. The *coverage* of a PI indicates the probability that the interval contains the actual value of the predicted variable. A PI is called *valid*, or calibrated, if the coverage probability for a new test point $(X_{n+1}, Y_{n+1}) \sim \pi$ is guaranteed to be equal or greater than the designed confidence level. PI's coverage guarantees are discussed in the following.

B. Marginal and Conditional Coverage

The PI's coverage guarantee can be defined on average over a set of test points (*marginal* coverage guarantee) or pointwise for any fixed value $X_{n+1} = x$ (*conditional* coverage guarantee) [7]. For a distribution-free marginal coverage guarantee, the probability that the PI covers the true test value Y_{n+1} must be at least $1 - \alpha$ on average over a random draw from any underlying distribution π:

$$\mathbb{P}\left\{Y_{n+1} \in \hat{C}_{\pi,n}(X_{n+1})\right\} \geq 1 - \alpha. \quad (1)$$

Conditional coverage is a much stricter definition compared to marginal coverage and, hence, harder to ensure. A PI satisfies conditional coverage on the $1 - \alpha$ level if

$$\mathbb{P}\left\{Y_{n+1} \in \hat{C}_{\pi,n}(X_{n+1})|X_{n+1} = x\right\} \geq 1 - \alpha \quad (2)$$

meaning that for any point x, the probability that $\hat{C}_{\pi,n}$ covers $X_{n+1} = x$ must be at least $1 - \alpha$. To demonstrate the difference between marginal and conditional coverage, [7] presents the following example:

> *Suppose that each data point i corresponds to a patient, with X_i encoding relevant covariates (age, family history, current symptoms, etc.), while the response Y_i measures a quantitative outcome (e.g., reduction in blood pressure after treatment with a drug). When a new patient arrives at the doctor's office with covariate values X_{n+1}, the doctor would like to be able to predict their eventual outcome Y_{n+1} with a range, making a statement along the lines of: "Based on your age, family history, and current symptoms, you can expect your blood pressure to go down by 10–15 mmHg".*

When setting $\alpha = 0.05$, the statement made by the doctor should hold with a probability of 95%. For marginal coverage, the statement has a 95% probability of being accurate on average for all possible patients. Since we consider the average, the statement might have a significantly lower, even 0%, chance of being accurate for patients of a specific age group, but is compensated by a coverage probability that is higher than 95% for the other age groups. On the other hand, for conditional coverage the statement made by the doctor must hold with 95% probability for every individual patient, regardless of age. Therefore, conditional coverage is more difficult to ensure.

Due to the stricter requirements, conditional coverage cannot be satisfied in distribution-free settings [8]. Consequently, most probabilistic forecasting methods focus on satisfying marginal coverage, or a compromise between marginal and conditional coverage. Please notice that methods ensuring marginal coverage can possibly, but not necessarily, obtain conditional coverage as well.

C. Ensemble Learning

Ensemble learning is a performance-enhancing technique for statistical and machine learning algorithms. In ensemble learning, a prediction model is built by using a collection of simpler base models [9], each one independently optimized to solve the same problem. In the context of machine learning, an ensemble model can be broadly defined as a system constructed with a set of individual models working in parallel and whose outputs are combined with a decision fusion strategy to produce a single answer for a given problem [10].

By combining a group of weak learners into one strong/expert learner, ensemble learning produces significantly improved results compared to individual learners [11]. The learners in the ensemble are often termed *member* learners. They can be any machine learning algorithm, such as neural networks, support vector machines, or decision trees [10]. The ensemble of learners can be generated using three different approaches:

- *heterogeneous* ensembles: the member learners come from several different classification or regression algorithms;
- *homogeneous* ensembles: member learners are generated using the same algorithm, but they are trained on different data;
- a combination of the two techniques.

The homogeneous ensemble method is formalized in Algorithm 1. The ensemble method is implemented in two steps.

Algorithm 1: Homogeneous Ensemble Learning

---

input : Data $\{(x_i, y_i)\}_{i=1}^{n}$, base learning algorithm A, aggregation function ϕ

output : Ensemble regression function $\hat{\mu}_\phi$

1 **for** $b = 1, \ldots, B$ **do**
2   Sample an index set $S_b = (i_{b,1}, \ldots, i_{b,m})$ from indices $(1, \ldots, n)$ with or without replacement
3   Compute $\hat{\mu}_b = A((X_{i_{b,1}}, Y_{i_{b,1}}), \ldots, (X_{i_{b,m}}, Y_{i_{b,m}}))$
4 Define $\hat{\mu}_\phi = \phi(\hat{\mu}_1, \ldots, \hat{\mu}_B)$

First, a population of B base learners is trained using different datasets, or different *multisets* sampled from the available training data. Then, the predictions of the base learners are combined to form a single predictor by using an aggregation function ϕ. Popular aggregation functions include the mean, median, or trimmed mean. Using different aggregation functions have different benefits, e.g. the mean reduces the mean square error (MSE), the median reduces the sensitivity to outliers, and the trimmed mean gives a compromise of both [8].

The multisets can be obtained from the training data by using several different methods, e.g. *bootstrapping* or *subsampling* [11]. Bootstrapping creates multiple samples from the original data by randomly sampling *with* replacement [9], where the sampled multisets often have the same size as the original dataset. Contrarily to bootstrapping, subsampling creates multisets from the original data by extracting subsets *without* replacement. One of the earliest and simplest ensemble method is bootstrap aggregating, or *bagging* for short. Bagging uses bootstrapping together with the mean aggregation function to create ensemble models. Subsample aggregating, or *subagging*, is a variant of bagging, where subsampling replaces bootstrapping.

Ensemble learners can produce very accurate results, since combining several models with relatively similar bias reduces the variance and improves the generalization capability [12]. However, this is only the case if the member learners are sufficiently diverse and accurate. Diversity between member learners is essential for the ensemble performance, since little is gained by combining a vast ensemble of learners if they all produce the same result. It is challenging to obtain diversity among the member learners since they are all optimized to solve the same task and, usually, they are trained on data derived from the same dataset, which makes the learners being highly correlated. There is a trade-off between the performance of the individual learners and the diversity among them. Combining very accurate but highly correlated learners often gives worse results than a combination of accurate and less accurate learners, since complementarity is more important than individual performance [13].

D. Conformal Prediction (CP)

CP is an on-top probabilistic forecasting framework that constructs marginally valid PIs based on a similarity measurement called *conformity* [14]. CP makes no hard distributional assumptions besides that observations must be exchangeable, i.e., the information provided by the observations is independent of the order in which the observations are presented. Despite this appealing feature, CP constructs PIs with constant or slightly varying length [15]. In addition, the exchangeability assumption makes CP unsuitable for time series data.

CP was first introduced as a transductive inference method [16], where different data realizations are presented several times to the underlying learning algorithm, making it unsuitable for models that during training iterate through the data samples until convergence [17]. The *inductive* conformal prediction, proposed by [18], avoids the shortcomings of the transductive CP method, but requires the training data to be split into two disjoint sets. From now on, we refer to the inductive method when mentioning CP.

For regression problems, CP starts from a training set $\{(X_i, Y_i)\}_{i=1}^n$ with pairs of predictors $X_i \in \mathbb{R}^d$ and response variables $Y_i \in \mathbb{R}$, and splits it into two subsets: the proper training set, $\mathcal{I}_1$, and a calibration set, $\mathcal{I}_2$. A regression model is fitted using $\mathcal{I}_1$, while the conformity score – a statistic of the prediction errors (residuals) obtained from $\mathcal{I}_2$ – is used to quantify the uncertainty in future predictions. Given a new observation $X_{n+1} = x$, we require that the conditional PI for the regressand Y_{n+1} with miscoverage rate α, denoted $\hat{C}_\alpha(x)$, must satisfy:

$$\mathbb{P}\left\{Y_{n+1} \in \hat{C}_\alpha(X_{n+1} = x)\right\} \geq 1 - \alpha. \quad (3)$$

CP provides this conditional PI as:

$$\hat{C}_\alpha(x) = [\hat{\mu}(x) - Q_{1-\alpha}(\mathcal{R}, \mathcal{I}_2), \hat{\mu}(x) + Q_{1-\alpha}(\mathcal{R}, \mathcal{I}_2)], \quad (4)$$

where $\hat{\mu}(x)$ is the prediction made by the underlying regression model, $\mathcal{R}$ is the set of residuals R_i computed from the predictions of the samples $i \in \mathcal{I}_2$, and the conformity score $Q_{1-\alpha}(\mathcal{R}, \mathcal{I}_2)$ is the $(1-\alpha)$-th quantile of $\mathcal{R}$ [15]. The residuals used to obtain the conformity score are often computed with the L_1 norm, but other distance measures can be used. From Eq. (4) it is clear that CP was designed with homoscedastic data in mind, since the PI is constructed as the conditional mean estimate of the response variable with a fixed-width band around it [19].

For an in-depth introduction to CP and its applications, we refer the interested reader to a recent tutorial [20].

E. Quantile Regression (QR)

QR aims at estimating a conditional quantile function (CQF) of Y given X at the specified α. The CQF is defined as:

$$q_\alpha(x) = \inf\{y \in \mathbb{R} : F_Y(y|X = x) \geq \alpha\}, \quad (5)$$

where $F_Y(y)$ is the conditional distribution function of Y, whose probability density function can be estimated from empirical CQFs with miscoverage in the range $(0 < \alpha < 1)$ [21], [22]. PIs can be obtained directly from two empirical CQFs computed from the training set. The confidence level $(1 - \alpha)$ of the PI is the difference between such two quantile levels. The estimated conditional PI of QR thus becomes:

$$\hat{C}_\alpha(x) = \left[\hat{q}_{\alpha_{lo}}(x), \hat{q}_{\alpha_{hi}}(x)\right], \quad (6)$$

where $\hat{q}_{\alpha_{lo}}(x)$ and $\hat{q}_{\alpha_{hi}}(x)$ are the empirical CQFs computed for $\alpha_{lo} = \alpha/2$ and $\alpha_{hi} = 1 - \alpha/2$.

Unlike the PI in Eq. (4), the width of the PI in Eq. (6) depends on each specific data point x and can vary significantly from point to point. Therefore, QR yields intervals that adapt to heteroscedasticity in the data. However, when the ideal interval $C_\alpha(x)$ is replaced by the finite sample estimate $\hat{C}_\alpha(x)$ in Eq. (6), the actual coverage of the PI is not guaranteed to match the designed confidence level $(1 - \alpha)$ [15].

The estimation of $\hat{q}_{\alpha_{lo}}(x)$ and $\hat{q}_{\alpha_{hi}}(x)$ can be cast as a optimization problem that minimizes the *pinball loss*. The pinball loss of an observation pair (x_i, y_i) is defined as:

$$L_{\alpha,i} = \begin{cases} (1 - \alpha)(\hat{q}_\alpha(x_i) - y_i), & \hat{q}_\alpha(x_i) \geq y_i; \\ \alpha(y_i - \hat{q}_\alpha(x_i)), & \hat{q}_\alpha(x_i) < y_i, \end{cases} \quad (7)$$

where y_i denotes the i-th sample response and $\hat{q}_\alpha(x_i)$ is the α-th quantile estimated for the corresponding predictor [23]. The pinball loss measures how well the estimated quantile relates to the actual distribution of the data: the lower the pinball loss, the more accurate is the estimation. The pinball loss can be used as the objective function to train a deep learning model [24], [25], [26]. We refer to a neural network (NN) that performs QR and is optimized with Eq. (7) as a *QRNN*.

F. Conformalized quantile regression (CQR)

CQR is a probabilistic forecasting method that combines CP and QR to construct valid PIs for heteroscedastic data [15], [27]. CQR inherits the advantages of both QR and CP: the properties of QR allow the method to adapt to the local variability in the data and the use of CP guarantees valid marginal coverage. Similarly to CP, CQR assumes the samples to be exchangeable and splits the training data into a proper training set and a calibration set. The resulting PIs are conformalized using the conformity scores

$$E_i = \max \left\{ \hat{q}_{\alpha_{lo}}(x_i) - y_i, y_i - \hat{q}_{\alpha_{hi}}(x_i) \right\}, \ i \in \mathcal{I}_2 \quad (8)$$

which quantify the error made by the PI of the stand-alone QR algorithm, as specified by the bounds $\hat{q}_{\alpha_{lo}}(x_i)$ and $\hat{q}_{\alpha_{hi}}(x_i)$ in Eq. (6).

The CQR PIs are calculated as follows:

$$\hat{C}_\alpha(x) = [\, \hat{q}_{\alpha_{lo}}(x) - Q_{1-\alpha}(\mathcal{E}, \mathcal{I}_2),$$
$$\hat{q}_{\alpha_{hi}}(x) + Q_{1-\alpha}(\mathcal{E}, \mathcal{I}_2) \,] \,, \quad (9)$$

where $\mathcal{E} = \{E_i\}_{i \in \mathcal{I}_2}$. Note that the value of $Q_{1-\alpha}(\mathcal{E}, \mathcal{I}_2)$, which is used to conformalize the PIs constructed by the QR algorithm, is fixed for all new data points x, similarly to $Q_{1-\alpha}(\mathcal{R}, \mathcal{I}_2)$ in CP.

The use of CQR in combination with QRNNs can yield unnecessarily wide PIs [15]. This problem can be addressed by tuning the nominal quantile levels, α_{lo} and α_{hi}, of the underlying QRNN as additional hyperparameters, which does not invalidate the coverage guarantee.

G. Ensemble Batch Prediction Intervals (EnbPI)

EnbPI is a method inspired by CP that builds distribution-free PIs for nonstationary time series [8]. EnbPI assumes a time series data generating process on the form:

$$Y_i = f(X_i) + \epsilon_i, \quad i = 1, 2, 3, \dots$$

The error process $\{\epsilon_i\}_{i \geq 1}$ is assumed to be stationary and strongly mixing, which replaces the exchangeability assumption required by CP. The probabilistic forecasts are constructed by aggregating point forecasts produced from leave-one-out predictions[1] constructed using homogeneous bootstrapped ensemble estimators. The diversity between the ensemble learners is obtained by training them on different subsets of the original training dataset.

The aggregated point predictions are used to build a PI with width equal to the $(1-\alpha)$-th empirical quantile of the latest T observed residuals. Given the training data $\{(x_i, y_i)\}_{i=1}^T$, the PI at time t is defined as:

$$\hat{C}_\alpha(x_t) = [\, \hat{f}_{-t}(x_t) - Q_{1-\alpha}(\epsilon_T),$$
$$\hat{f}_{-t}(x_t) + Q_{1-\alpha}(\epsilon_T) \,] \,, \quad (10)$$

where $\epsilon_T = \{\hat{\epsilon}_i\}_{i=t-1}^{t-T}$. The residuals used to obtain the conformity score are computed as the absolute error between the training sample responses and the predictions of the leave-one-out estimators, denoted $\hat{f}_{-t}(x_t)$.

---

[1]When dealing with time series data, future data points are those left outside (i.e., leave-future-out).

To handle nonstationary time series data, EnbPI exploits a sliding window of size s to account for new data without refitting the underlying regression algorithm. In particular, the list containing the T out-of-sample residuals is updated every s predictions. This allows the width of the subsequent PIs to vary and makes the calibration of the PI widths more dynamic and accurate. The ensemble learners are only trained once and are used to predict the center of the PIs for the future time steps. Hence, the ensemble learners are assumed to model f sufficiently well. In practice, this assumption can fail when the window size s is large and long-term predictions are made. Indeed, if the dynamics of nonstationary time series significantly change over time, the original model f will eventually stop to describe well the underlying process. Valid coverage can still be obtained if a small window size is used, but the resulting intervals can become inflated if the out-of-sample absolute residuals are large.

III. THE EnCQR ALGORITHM

The proposed *ensemble conformalized quantile regression* (EnCQR) algorithm, summarized in Alg. 2, combines an ensemble of QR learners with CP to construct PIs for time series. EnCQR consists of three main steps:

1) **Train the ensemble learners (lines 1-9)**. The homogeneous learners are trained on independent subsets constructed in lines 1-4. Next, the ensemble learners are used to construct leave-one-out estimates for each observation i, by aggregating all the learners trained on subsets not including sample i (line 7). Then, the conformity scores between the aggregated leave-one-out predictions and the training labels are computed (line 8).

2) **Sequentially construct PI for the observations in the test set (lines 10-15)**. The observations in the test set are predicted using the ensemble learners, which produce a set of B quantile functions for both the upper and lower PI limit. The final PI limits are obtained by first aggregating the estimated quantile functions and then by conformalizing them using the $(1-\alpha)$-th quantile of the out-of-sample residuals calculated during training.

3) **Update the residuals (lines 16-20)**. The out-of-sample residuals are updated after every s new observations are predicted by replacing the oldest s elements of the list, such that its length remains the same.
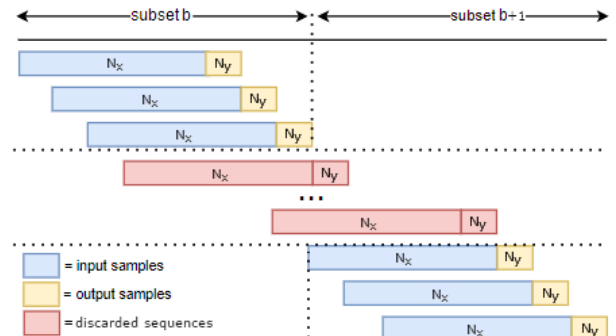

Fig. 1. Data split for training of ensemble learners.

Algorithm 2: EnCQR

---

input : Training data $\{(x_i, y_i)\}_{i=1}^{T}$, quantile regression algorithm A, confidence level $\alpha \in (0,1)$, aggregation function ϕ, no. of ensemble models B, window size s, and test data $\{(x_i, y_i)\}_{i=T+1}^{T+T'}$, where $y_i, i = T+1, \ldots, T+T'$ become available only after the batch of s PIs is constructed.

output : Ensemble PIs $\{\hat{C}_{\alpha, T}(x_i)\}_{i=T+1}^{T+T'}$

1 Determine length of ensemble subset: $T_b \leq T/B$

2 for $b = 1, \ldots, B$ do

3   Sample index set $S_b = (i_{T_b \times b}, \ldots, i_{T_b \times b + T_b})$ from indices $(1, \ldots, T)$

4   Fit quantile ensemble estimator $[\hat{q}_{\alpha_{lo}}^{(b)}, \hat{q}_{\alpha_{hi}}^{(b)}] = A(\{(x_i, y_i)\}_{i \in S_b})$

5 Initialize $\mathcal{E}_{lo} = \{\}$, $\mathcal{E}_{hi} = \{\}$

6 for $i = 1, \ldots, T$ do

7 $[\hat{q}_{\alpha_{lo}}(x_i), \hat{q}_{\alpha_{hi}}(x_i)] =$ $[\phi(\{\hat{q}_{\alpha_{lo}}^{(b)}(x_i)\}_{b \in \mathcal{B}_{-i}}), \phi(\{\hat{q}_{\alpha_{hi}}^{(b)}(x_i)\}_{b \in \mathcal{B}_{-i}})]$ with $\mathcal{B}_{-i} = \{S_b \text{ s.t. } i \notin S_b\}$

8   Compute E_{lo_i} and E_{hi_i} as in Eq. (11)

9 $\mathcal{E}_{lo} = \mathcal{E}_{lo} \cup \{E_{lo_i}\}, \mathcal{E}_{hi} = \mathcal{E}_{hi} \cup \{E_{hi_i}\}$

10 for $i = T+1, \ldots, T+T'$ do

11   Let $\hat{q}_{\alpha_{lo}}(x_i) = \phi(\{\hat{q}_{\alpha_{lo}}^{(b)}(x_i)\}_{b=1}^{B})$

12   Let $\hat{q}_{\alpha_{hi}}(x_i) = \phi(\{\hat{q}_{\alpha_{hi}}^{(b)}(x_i)\}_{b=1}^{B})$

13   Let $\omega_{lo_i} = (1-\alpha)$-th quantile of $\mathcal{E}_{lo}$

14   Let $\omega_{hi_i} = (1-\alpha)$-th quantile of $\mathcal{E}_{hi}$

15   Return $\hat{C}_{\alpha, T}(x_i) = [\hat{q}_{\alpha_{lo}}(x_i) - \omega_{lo_i}, \hat{q}_{\alpha_{hi}}(x_i) + \omega_{hi_i}]$

16   if $i - T = 0 \mod s$ then

17   for $j = i - s, \ldots, i - 1$ do

18   Compute E_{lo_j} and E_{hi_j} as in Eq. (11)

19 $\mathcal{E}_{lo} = (\mathcal{E}_{lo} - \{E_{lo_{:s}}\}) \cup \{E_{lo_j}\}$ and reset index of $\mathcal{E}_{lo}$

20 $\mathcal{E}_{hi} = (\mathcal{E}_{hi} - \{E_{hi_{:s}}\}) \cup \{E_{hi_j}\}$ and reset index of $\mathcal{E}_{hi}$

Both EnbPI and EnCQR use homogeneous ensembles, but they differ in how training data subsets for the ensemble learners are built: EnbPI uses bootstrap samples drawn with replacement, while EnCQR creates B disjoint subsets of length T_b, each used to train one learner (see Fig. 1). The trained learners can be used to produce out-of-sample residuals when applied to data that are not in the subset used for training. Within each subset, multiple overlapping input-output sequences (in blue-yellow) are extracted to be used as batches in the training of one learner. These have length $N = N_x + N_y \leq T_b$, where N_x and N_y denote the length of the input/output sequences. All sequences that go across subsets (in red) must be discarded for the residuals to remain valid. A large N_x let learners capture long time dependencies in the data, while a large B results in more learners and more robust predictions. EnCQR benefits from large datasets (length

T), as the number of residuals to perform CP is $T - B \cdot N_x$. When T is small, N_x and B should be reduced, trading the amount of residuals with the performance of the ensemble model.

Similarly to EnbPI, EnCQR utilizes an out-of-sample ensemble prediction strategy to apply CP to time series data, but replaces the aggregated point prediction of EnbPI with aggregated quantile predictions for the upper and lower bound of a PI. As in CQR, the aggregated PI obtained using the ensemble learners are *conformalized* using asymmetrical conformity scores defined as:

$$E_{lo_i} = \hat{q}_{\alpha_{lo}}(x_i) - y_i, \quad E_{hi_i} = y_i - \hat{q}_{\alpha_{hi}}(x_i). \quad (11)$$

The motivation for defining asymmetric conformity scores is that the distribution of the conformity scores for the two estimated conditional quantile functions can be skewed, resulting in the PI coverage error being asymmetrically spread over the left and right tails. If this occurs, the intervals can be wrongly conformalized, resulting in coverage below the target level. The asymmetric conformity score solves this problem by controlling the coverage of the two quantile functions independently. It can be shown that by using the asymmetric conformity score it is possible to obtain a stronger coverage guarantee compared to the original formulation, at the possible expense of slightly wider intervals [15].

Substitution of the point prediction in Eq. (10) with the estimated quantile functions and the absolute conformity score with the asymmetric conformity score yields the EnCQR PI, which is defined as follows:

$$\hat{C}_{\alpha, \phi}(x_t) = [\hat{q}_{\alpha_{lo}}(x_t) - Q_{1-\alpha}(\mathcal{E}_{lo}), \\ \hat{q}_{\alpha_{hi}}(x_t) + Q_{1-\alpha}(\mathcal{E}_{hi})], \quad (12)$$

where $\mathcal{E}_{lo} = \{E_{lo_i}\}_{i=t-1}^{t-T}$ and $\mathcal{E}_{hi} = \{E_{hi_i}\}_{i=t-1}^{t-T}$. As in CQR, the nominal quantile levels estimated by the underlying QR algorithm in EnCQR can be tuned to construct possibly PIs.

EnCQR combines the principles of EnbPI and CQR, while tackling the individual shortcomings of both methods. Specifically, EnCQR replaces the symmetric intervals in the EnbPI method with locally adaptive PIs by utilizing QR, as in CQR. Furthermore, EnCQR exploits the same moving window approach of EnbPI, by updating every s predictions the conformity scores used to conformalize the QR intervals. This allows to take into account new observations as they become available. In addition, the update of the conformity scores and sequential construction of the PI makes EnCQR particularly suited for heteroscedastic time series.

IV. EXPERIMENTS

The experiments focus on comparing the PI obtained by EnCQR with methods based only on QR or CP, which use the same underlying regression algorithm. To demonstrate the versatility of EnCQR we consider regression algorithms based both on deep learning architectures and traditional machine learning approaches. Finding the best underlying regression algorithm for the specific dataset can improve the performance of both EnCQR and the competing methods, but not their *relative difference* in performance. Therefore, finding such an

optimal algorithm is outside the scope of our evaluation. The software implementation of EnQCR is available online[2]

A. Datasets

To test the effectiveness and generality of EnCQR in quantifying uncertainty in time series forecasting, we consider five real-world time series where the amount of heteroscedasticity and the seasonal patterns are significantly different. We also considered the presence of exogenous variables as additional input time series, i.e., a multivariate input – univariate output setting. All the time series are partitioned into three disjoint sets, training, test, and validation, and are independently normalized by scaling the values to lie in the interval [0,1]. The time series from the first four datasets have hourly resolution and are reshaped into input-output pairs of size 168-24 (1week-1day) with a sliding window. In the last dataset, the input-output pairs have sizes 24-6.

The **Portugal** dataset[3] consists of time series of electricity consumption from 270 customers located in Portugal. The data have an hourly resolution from 2012 to 2014 and the consumption is measured in kiloWatt (kW). We arbitrarily selected 5 among the 270 time series. For each time series, the training/validation/test split is 12/12/12 months.

The **Elvia** dataset is shared by Elvia AS, a distribution system operator that operates grids in the Norwegian counties of Oslo, Viken and Innlandet. The dataset describes the electricity consumption for three classes of end-users; industry, household, and cabin. The data consists of two time series recorded hourly from 1 June 2018 to 1 June 2020 that contains the observed electricity load and the temperature forecast. One time series from each user category was arbitrarily selected. The training/validation/test split is 12/6/6 months (the 1st year is used for training, while odd and even months of the 2nd year are used as validation and test data, respectively).

The **Solar** and **Wind** datasets[4] contain solar and wind power production data from Webberville Solar Farm and Hackberry wind farm in Texas, US. Data is recorded in MWh per hour from January 2017 to January 2020. Both datasets contain 6 ambient climate features. For both datasets, the training/validation/test split is 12/12/12 months.

The **Temperature** dataset[5] contains weather data collected by the Max Planck Institute for Biogeochemistry. The dataset contains 15 meteorological variables, among which the ambient temperature, that is collected every 10 minutes from January 2009 to January 2016. The dataset is subsampled to have an hourly resolution. The first 80% of the dataset is used for training, 10% for validation, and 10% for testing.

The two power consumption and, in particular, the solar production and temperature time series are characterized by a strong seasonal pattern, which is not present in the wind power time series. To measure the heteroscedasticity in each time series, we first calculate the standard deviation (std) of all

values observed at a given hour of day over the entire dataset, and then the std of the resulting 24 values:

$$\mathrm{std}\left(\mathrm{std}\{h_1 d_1, h_1 d_2, \ldots\}, \ldots, \mathrm{std}\{h_{24} d_1, h_{24} d_2, \ldots\}\right),$$

where, e.g., $h_1 d_2$ is the measurement at hour 1 of day 2. A higher value indicates more variability between different hours, i.e. a higher degree of heteroscedasticity. Tab. I show a significant difference between the time series: the Portugal time series 250 has the lowest degree of variability, while the Solar and Temperature time series have the largest ones.

TABLE I
MEASURE OF HETEROSCEDASTICITY FOR EACH DATASET.

	Time series	Variability		Time series	Variability
Portugal	ID: 250	$1.0e^{-4}$	Elvia	Industry	$9.3e^{-4}$
	ID: 77	$4.3e^{-4}$		Household	$1.9e^{-4}$
	ID: 50	$2.1e^{-4}$		Cabin	$1.9e^{-4}$
	ID: 90	$1.3e^{-4}$		Solar	$21.0e^{-4}$
	ID: 27	$4.2e^{-4}$		Wind	$3.0e^{-4}$
				Temperature	$39.1e^{-4}$

B. Experimental Setup

The experimental details are briefly summarized here. Further details are given in the Appendix.

a) Reference models: We compare EnCQR with EnbPI and QR. All methods are based on the same underlying regression algorithms. Additionally, we include the SARIMA model as a representative of traditional statistical models for time series forecasting. SARIMA is meant to be a baseline rather than a competitor of EnCQR. As it cannot model heteroscedasticity, it produces PIs that are not adaptive and are valid only under restrictive assumptions.

All models are designed to construct multi-step probabilistic forecasts in the form of 90% PIs, i.e. $\alpha = 0.10$.

b) Regression algorithms: We perform experiments with quantile random forest (QRF) and QRNNs as the underlying regression algorithms. Given the prominence of NNs in recent time series forecasting research, we decided to use two different QRNN architectures: Temporal Convolutional Network (TCN) [28], and Long Short-term Memory Neural Network (LSTM) [29].

c) Model training: We conduct a random search over the NN hyperparameters, detailed in the Appendix. All NNs are trained to minimize the pinball loss. The ensemble NNs in EnbPI are trained to predict only the 0.5 quantile. The QRNN and EnCQR models are trained to predict, at the same time, multiple quantile levels, 0.95, 0.50 and 0.05, rather than fitting individual network instances for each quantile. This is done by modifying the pinball loss in (7), which is averaged over all observations and all target quantiles. Early stopping terminates the training process if the pinball loss on the validation set does not improve over 50 consecutive epochs. For the QRF models, we build 10 trees that are expanded until all leaves are pure or until all leaves contain less than 2 samples, and we estimate the 0.95, 0.50 and 0.05 quantiles.

TABLE II
PICP, PINAW, AND CWC SCORES FOR ENCQR, QR, AND ENBPI APPLIED TO DIFFERENT REGRESSION MODELS. WE REPORT MEAN (AND STANDARD DEVIATION) OVER 10 RUNS FOR NN-BASED AND RF-BASED MODELS AND A SINGLE RUN FOR THE DETERMINISTIC SARIMA MODEL.

			LSTM			TCN			Random forest			SARIMA		
		Model	PICP	PINAW	CWC	PICP	PINAW	CWC	PICP	PINAW	CWC	PICP	PINAW	CWC
Portugal dataset	ID: 250	EnCQR	.884 (.011)	.210 (.011)	**.783**	.890 (.010)	.210 (.015)	**.787**	.900 (.003)	.219 (.002)	.781			
		QR	.675 (.030)	.127 (.032)	.191	.718 (.068)	.165 (.017)	.309	.696 (.016)	.167 (.013)	.239	.789	.154	.584
		EnbPI	.880 (.005)	.217 (.012)	.773	.890 (.013)	.238 (.011)	.759	.876 (.002)	.202 (.001)	**.784**			
	ID: 77	EnCQR	.900 (.021)	.249 (.022)	.751	.900 (.010)	.197 (.009)	**.802**	.903 (.003)	.215 (.002)	**.784**			
		QR	.739 (.063)	.134 (.028)	.397	.618 (.044)	.144 (.013)	.078	.589 (.019)	.124 (.005)	.048	.726	.179	.331
		EnbPI	.901 (.008)	.297 (.026)	.702	.895 (.017)	.273 (.020)	.726	.926 (.002)	.227 (.002)	.757			
	ID: 50	EnCQR	.890 (.014)	.252 (.009)	**.745**	.900 (.029)	.326 (.027)	**.673**	.911 (.002)	.285 (.002)	.712			
		QR	.838 (.062)	.242 (.055)	.675	.758 (.011)	.211 (.005)	.430	.766 (.007)	.193 (.002)	.470	.802	.219	.585
		EnbPI	.905 (.008)	.299 (.034)	.700	.897 (.003)	.344 (.023)	.655	.913 (.002)	.275 (.002)	**.721**			
	ID: 90	EnCQR	.900 (.021)	.263 (.016)	**.737**	.912 (.023)	.305 (.025)	.692	.914 (.004)	.295 (.003)	.700			
		QR	.610 (.032)	.153 (.009)	.067	.816 (.042)	.232 (.024)	.621	.713 (.015)	.195 (.005)	.281	.893	.215	**.783**
		EnbPI	.900 (.005)	.331 (.011)	.669	.926 (.006)	.288 (.017)	**.697**	.900 (.002)	.285 (.002)	**.715**			
	ID: 27	EnCQR	.910 (.015)	.225 (.018)	**.772**	.890 (.012)	.229 (.015)	**.768**	.911 (.003)	.189 (.002)	**.808**			
		QR	.800 (.125)	.184 (.061)	.604	.898 (.021)	.240 (.025)	.759	.790 (.007)	.147 (.002)	.593	.590	.196	.044
		EnbPI	.900 (.002)	.264 (.014)	.736	.895 (.005)	.277 (.027)	.722	.916 (.002)	.200 (.002)	.793			
Elvia dataset	Industry	EnCQR	.900 (.010)	.296 (.023)	.704	.900 (.019)	.303 (.009)	**.697**	.968 (.003)	.325 (.005)	.587			
		QR	.923 (.010)	.241 (.017)	**.747**	.927 (.039)	.293 (.039)	.691	.847 (.017)	.187 (.005)	**.747**	.887	.430	.567
		EnbPI	.927 (.005)	.486 (.021)	.502	.960 (.006)	.347 (.048)	.586	.979 (.001)	.339 (.007)	.548			
	Household	EnCQR	.909 (.021)	.242 (.032)	**.756**	.892 (.035)	.414 (.041)	.584	.949 (.003)	.457 (.006)	**.505**			
		QR	.900 (.023)	.279 (.072)	.721	.904 (.015)	.360 (.032)	**.639**	.764 (.008)	.157 (.004)	.484	.894	.148	**.851**
		EnbPI	.980 (.006)	.369 (.033)	.520	.973 (.004)	.445 (.018)	.473	.998 (.001)	.513 (.010)	.365			
	Cabin	EnCQR	.910 (.022)	.259 (.027)	**.738**	.906 (.028)	.401 (.030)	**.598**	.933 (.004)	.371 (.004)	**.609**			
		QR	.903 (.057)	.275 (.072)	.724	.861 (.055)	.447 (.060)	.528	.822 (.009)	.204 (.006)	.608	.930	.257	.723
		EnbPI	.943 (.008)	.303 (.016)	.659	.960 (.005)	.574 (.024)	.382	.972 (.002)	.391 (.004)	.521			
Solar		EnCQR	.900 (.007)	.354 (.011)	**.646**	.900 (.009)	.459 (.028)	.540	.917 (.002)	.310 (.005)	**.684**			
		QR	.910 (.011)	.365 (.018)	.633	.840 (.046)	.334 (.029)	**.597**	.863 (.004)	.298 (.005)	.673	.842	.543	.413
		EnbPI	.910 (.003)	.631 (.023)	.367	.923 (.006)	.789 (.019)	.207	.906 (.001)	.622 (.005)	.377			
Wind		EnCQR	.915 (.009)	.859 (.025)	.140	.913 (.004)	.897 (.015)	.102	.906 (.002)	.799 (.003)	**.200**			
		QR	.894 (.004)	.713 (.012)	**.286**	.775 (.035)	.682 (.051)	**.198**	.744 (.008)	.598 (.006)	.193	.980	1.00	.000
		EnbPI	.901 (.002)	.974 (.005)	.025	.913 (.007)	1.00 (.033)	.000	.900 (.002)	.895 (.006)	.104			
Temperature		EnCQR	.942 (.010)	.268 (.018)	**.692**	.944 (.009)	.279 (.055)	**.680**	.943 (.011)	.337 (.078)	.621			
		QR	.820 (.031)	.198 (.019)	.611	.684 (.105)	.170 (.031)	.270	.931 (.060)	.365 (.081)	.555	.813	.395	.482
		EnbPI	.940 (.008)	.341 (.022)	.627	.932 (.004)	.335 (.056)	.645	.941 (.012)	.305 (.073)	**.658**			

TABLE III
AVERAGE CWC ACROSS ALL DATASETS AND REGRESSION ALGORITHMS.

	EnCQR	QR	EnBPI	SARIMA
Avg. CWC	**.648**	.477	.558	.486

d) Ensemble learners: Based on the size of the datasets, we chose $B = 3$ to be a suitable number of ensemble learners in EnbPI and EnCQR. While using more learners could give better performance, for a fixed amount of data each of the B subsets would become smaller (see Fig. 1). As a consequence, the learners would be trained on less data and might not capture longer temporal dependencies.

The window size parameter s should reflect the nature of the data. Since all time series have hourly resolution and a strong daily seasonality, we set $s = 24$. Finally, all ensemble models use the mean as aggregation function ϕ.

e) Evaluation metrics: To evaluate the quality of the PIs, both their coverage and width must be quantified. For this we use two measures; the *prediction interval coverage probability* (PICP) [30]:

$$\text{PICP} = \frac{1}{n} \sum_{i=1}^{n} c_i, \quad c_i = \begin{cases} 1, & y_i \in [L_i, U_i] \\ 0, & y_i \notin [L_i, U_i] \end{cases}$$

and *prediction interval normalized average width* (PINAW):

$$\text{PINAW} = \frac{1}{nR} \sum_{i=1}^{n} (U_i - L_i), \quad R = y_{\max} - y_{\min}.$$

Here U and L denote the upper and lower bound of the PI. PICP alone is not sufficient to measure performance, since very wide PIs have high coverage but are less informative. An optimal PI has a PICP close to the designed confidence level and minimizes, at the same time, the PINAW. To summarize with a single value the quality of the PI, we adopt a modification of the *coverage width-based criterion* (CWC) [31] that penalizes under- and overcoverage in the same way:

$$\text{CWC} = (1 - \text{PINAW})e^{-\eta(\text{PICP}-(1-\alpha))^2}, \quad (13)$$

where η is a user-defined parameter that balances the PINAW and PICP contributions. In our experiment, we set $\eta = 30$.

C. Results and Analysis

Table II summarizes the results obtained on each dataset using the NN-based and RF-based models. We also report the results of SARIMA for comparison. For each dataset and regression algorithm, we highlight in bold the best result in terms of CWC. Table III reports the mean CWC of the different approaches across all datasets and regression algorithms.

1) PI Coverage and PI Width: Both EnCQR and EnbPI successfully construct approximately valid PIs for all time series, demonstrating that they work well for different data distributions, regardless of the regression algorithm used in the ensemble. However, the width of the PIs constructed by EnbPI considerably varies compared to EnCQR, which produces the sharpest valid PIs. This can be explained by referring to the variability measures presented in Table I; For time series with a low degree of heteroscedasticity, the quality of the PIs constructed by EnCQR and EnbPI is approximately equal, whereas for the more heteroscedastic time series, EnCQR constructs significantly sharper and more informative PIs. This is illustrated in Fig. 2, which plots PICP against PINAW for the NN- and RF-based models on the Solar and Portugal datasets. For both datasets, nearly all EnCQR instances in Fig. 2 are
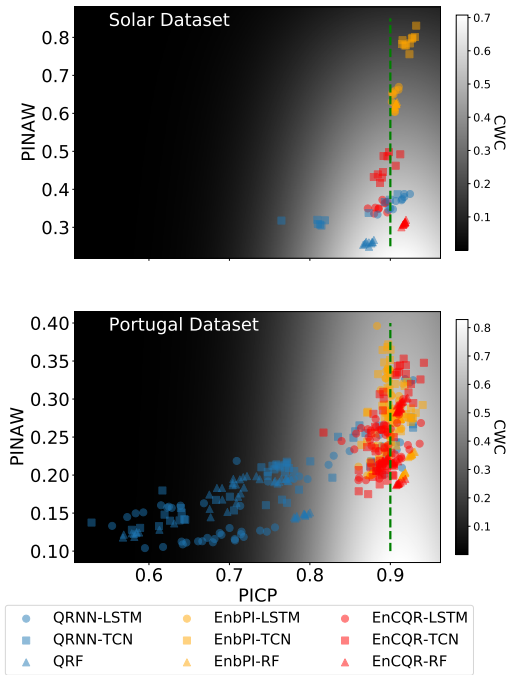

Fig. 2. PICP versus PINAW for all instances of the NN an RF models for the Solar and Portugal datasets. The background shows the value of CWC at each region: the brighter the better.

located in the bottom-right corner, indicating that these models produce PIs with the highest PICP and narrowest PINAW. For the Portugal dataset, EnbPI achieves a similar coverage level as EnCQR, but the PINAW in EnbPI is generally higher, indicating that the PIs are wider and less informative. For the Solar dataset, which is more heteroscedastic, the difference in PINAW between the EnbPI and EnCQR is even larger.

SARIMA, QRF, and QRNN yield PIs whose PICP greatly varies in different datasets, as they lack robustness and do not enjoy the coverage guarantee of the CP-based models. In fact, they might construct very narrow intervals where the actual coverage of the PIs is significantly lower than the desired confidence level.

The results discussed so far are aligned with the CWC

values in Table II. Table III shows that when CWC is averaged across all dataset and regression algorithms EnCQR achieves the top performance and EnBPI, which comes second, achieves a significantly lower average CWC score. In the next analyses we focus only on EnCQR and EnbPI, since SARIMA, QRF and QRNN often fail to produce a valid coverage.

2) Symmetric vs. Adaptive PIs: Here, we study how producing symmetric PIs rather than locally adaptive PIs affects the quality of the results. The analysis is performed using the *Solar* time series, which has a strong seasonal pattern since most of the energy is produced at the central hours of the day. The variation in hourly energy production in Fig. 3, clearly indicates the presence of heteroscedasticity in the time series.

As previously stated, both EnCQR and EnbPI use a sliding window of size $s = 24$ to reflect the data collection process. As a result, the set of leave-one-out residuals is updated after every 24 hours. Consequentially, the width of the PIs generated by EnbPI is fixed for the 24-hours interval, indicating an equal amount of uncertainty for all the hours in a day. On the other hand, the adaptive EnCQR intervals show that there is significantly lower uncertainty during the night. The advantage of locally adaptive intervals for heteroscedastic data is clear: EnCQR constructs significantly more informative intervals for the hours with less variability.

In EnbPI to guarantee valid coverage with symmetric intervals when the data is heteroscedastic, the length of the intervals must increase significantly to include points far from the mean value. The length increases identically in both directions, which is undesirable if the spread from the mean is not symmetric. This is the case of the Solar time series: in the middle of the day, the deviation from the mean is greater in the downwards direction due to the potential absence of sun. On the other hand, the PI of EnCQR are not constrained to be symmetric and can, therefore, capture in which direction the variability is larger. Referring to Fig. 3, EnCQR-LSTM correctly captures the variability in the time series: the predicted 0.50 quantile is almost perfectly aligned with the median of the boxes, while the lower PI bound extends downwards much further than the upper PI bound.

3) The Effect of Conformalization: In EnCQR, the PI constructed by the ensemble of QR learners is conformalized by adding or subtracting an error term to the interval's width. This error term quantifies the accuracy of the original interval and addresses both under- and overcoverage, since the PIs can be extended or shortened to improve both PI coverage and width. To analyze the effect of conformalization, we use the results of the EnCQR-LSTM model for Station 77 from the Portugal dataset, the Industry user in the Elvia dataset, and the Solar dataset. Fig. 4 depicts the conformalized and original intervals of the time series associated with station 77 and clearly shows the improvement of conformalization. The original PIs do not cover the boxes of the underlying boxplot, which extends from the 1st to the 3rd quartile, hence indicating a significant undercoverage. The conformalization extends both the upper and lower PI bound and guarantees a valid coverage. The results for the other datasets are summarized in Table IV.
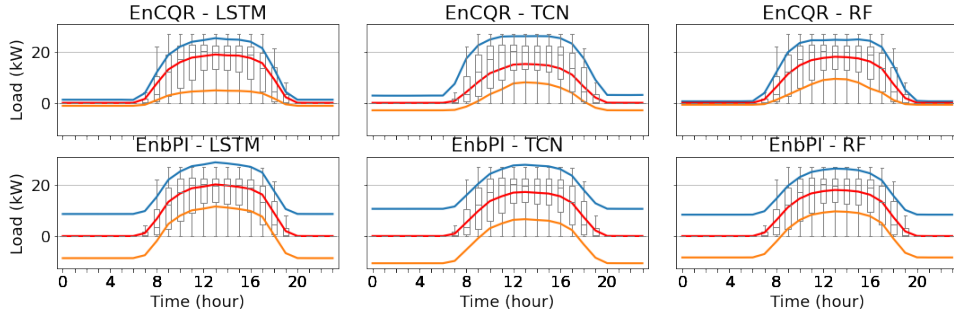
Fig. 3. Average PIs for all hours in the solar test dataset. The red line represent the predicted mean, and the blue and yellow line represent the average upper and lower PI bounds, respectively. All lines show the aggregated results from ten individual runs. The underlying boxplot represent the hourly variation for the observations in test datasets.
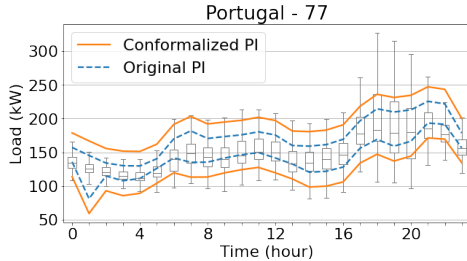

Fig. 4. Average original and conformalized PIs for all hours in the Portugal - 77 test set. The PICP of the original PI is improved by the conformalization.

TABLE IV
PICP AND PINAW FOR THE ORIGINAL AND CONFORMALIZED PIS FOR THE ENCQR-LSTM MODELS.

Time series	PICP / PINAW	
	conformalized PI	original PI
Portugal - 77	0.900 / 0.249	0.524 / 0.109
Elvia - Industry	0.900 / 0.296	0.550 / 0.157
Solar	0.900 / 0.354	0.724 / 0.272

V. DISCUSSION AND CONCLUSIONS

QR-based models tend to produce overly confident and often invalid PIs, since they are too narrow and the actual observations fall, on average, outside the PI boundaries more often than the specified confidence level. This lack of robustness motivates the need of probabilistic frameworks, such as CP, to obtain valid PIs. The recently proposed EnbPI method allows to apply CP to time series data. However, despite of the advantage of valid coverage, CP tends to be unnecessarily conservative as it constructs PIs with constant length, which are uninformative especially when dealing with heteroscedastic data.

In this paper we proposed EnCQR, a probabilistic time series forecasting method that leverages CP to generate PIs with valid coverage and ensemble learners performing QR to handle heteroscedastic data. Experiments on real-world datasets with different degrees of heteroscedasticity demonstrated the superior performance of the proposed method compared to methods based only on CP or QR. Our method outperforms CP-based models such as EnbPI in terms of PI sharpness and QR-based models in terms of PI coverage. For homoscedastic data, EnCQR performs approximately equal to EnbPI in terms

of PI quality. For heteroscedastic data, EnCQR outperforms EnbPI as the PI width adapt well to local variability.

The most appealing property of EnCQR is unarguably that the PIs are guaranteed to marginally satisfy the designed coverage rate for finite samples and are adaptive to local variability, hence construing sharper PIs compared to other CP-based methods. EnCQR is particularly suitable for large datasets, due to the need to create independent subsets of consecutive data. To have enough ensemble learners, while ensuring that they are trained on enough data and capture long-range dependencies, longer time series are warranted.

An advantage of EnCQR is that it can be applied on top of any ensemble of QR models, such as the two "standard" neural network architectures (LSTM and TCN) trained with the pinball loss. It is worth mentioning that several advanced deep learning models for probabilistic forecasting have been proposed in the past few years [32] and some of them are readily available in open-source libraries [33]. Despite producing accurate forecasts, none of these methods can generate PIs that are both valid and adaptive. Therefore, an interesting future work would be to replace the LSTM and TCN backbones with more powerful models to obtain narrower, yet valid and adaptive, PIs.

As a final remark, this paper focused on frequentist approaches to compute PIs, but uncertainty and interval quantification can also be estimated using Bayesian approaches. Bayesian uncertainty estimates often fail to capture the true data distribution, due to model bias that does not assign the right probability to every credible interval [34]. On the other hand, Bayesian methods are more robust when the data available are scarce and the model is very uncertain about the prediction. A principled and fair comparison between Bayesian and the proposed frequentist intervals must be done with caution and it might be interesting to explore in future work.

APPENDIX

A. Neural Networks implementation details

The neural network-based models are implemented in Tensorflow [35], [36]. The optimal network hyperparameters, such as learning rate, batch size, and layer units, are identified by performing a random hyperparameter search. More specifically, we randomly select different parameter configurations from specified intervals and we select the configuration that achieves

the highest performance (in terms of quality of the prediction interval) on the validation dataset.

All neural networks are trained using the Adam optimizer [37]. L2 regularization is used to prevent overfitting and improve the generalization capabilities of the network. The same L2 penalty regularization is applied to the input, hidden and output weights. For all networks, the λ_2 value (which specifies the contribution to the loss of the term that penalizes the L2 norm of the weights) is optimized during the hyperparameters search. In particular, the value of λ_2 is randomly sampled from the interval $[0, 0.1]$ using a logarithmic scale. The configurations of the specific deep learning models are described in the following.

1) TCN: The TCN setup, inspired by the DeepTCN[6] network presented by [28], consists of several stacked residual blocks containing dilated convolutional layers, followed by a final fully-connected layer that maps the output of the residual blocks into quantile predictions. The residual blocks, illustrated in Fig. 5, consist of two identical dilated causal convolutional layers, both followed by a batch normalization layer and ReLU activation. Depending on the number of residual blocks in the network, the residual block's output is either passed as input to the next residual block or to the final fully-connected output layer. Contrarily to the skip connections in ordinary residual networks, the skip connections in the TCN residual blocks contain a 1×1 convolutional layer with the same number of filters of the convolutional layers in the residual block, prior to the element-wise sum operation $\oplus$. The additional convolutional layer ensures that the sum operator receives tensors of the same shape, as the input and output of the TCN residual block can have different widths [38].
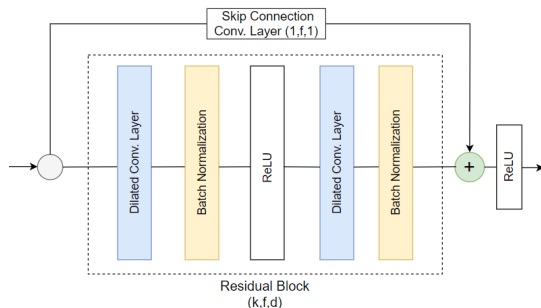

Fig. 5. TCN residual block, containing two identical dilated convolutional layers with kernel size = k, no. convolutional filters = f, and dilation factor = d. The skip connection consists of a convolutional layer with k = 1, d = 1, and the same number of filter as the convolutional layers within the residual block, followed by an element-wise addition $\oplus$

The optimal values of the TCN hyperparameters are searched within the following ranges:

- Dilation factor $d = 2^i, i \in [0, 5]$
- Kernel size $k = [2, 7]$
- No. filters $N_f = [2, 200]$
- Learning rate $\eta = [0.0001, 0.01]$
- L2 regularization parameter $\lambda_2 = [0.0001, 0.1]$
- Batch size $B = [16, 32, 56, 64]$

[6]https://github.com/oneday88/deepTCN

- Quantile ranges from the EnCQR models:
  $\hat{q_{lo}} = [0.01, 0.2], \hat{q_{hi}} = [0.7, 0.99]$

The optimal hyperparameter values found are reported in Tab.V

2) LSTM: The LSTM network contains one or more hidden LSTM layers, followed by a fully-connected output layer used to map the output from the LSTM layers into the actual quantile forecasts. All hidden layers have the same number of units. When stacking several LSTM layers, the output from the first LSTM layer is used as input to the next. Stacking several recurrent levels allows each recurrent layer to operate at different timescales [39], which often improves the network performance in sequence prediction problems.

Tensorflow provides both stateful and stateless LSTM cells[7]. In a stateful LSTM, the last state for each sample at index i in a batch will be used as initial state for the sample of index i in the following batch. In a stateless LSTM layer, the hidden states are reset after each batch and, therefore, the network cannot learn time dependencies spanning across different batches. The LSTM networks implemented in the experiments are stateful and the batch size is, therefore, fixed for all LSTM networks.

The optimal values of the LSTM hyperparameters are searched within the following ranges:

- No. units in hidden layers $N_u = [8, 200]$
- No. hidden layers $N_h = [1, 3]$
- Learning rate $\eta = [0.0001, 0.01]$
- L2 regularization parameter $\lambda_2 = [0.0001, 0.1]$
- Quantile ranges from the EnCQR models:
  $\hat{q_{lo}} = [0.01, 0.2], \hat{q_{hi}} = [0.7, 0.99]$

The optimal hyperparameter values are reported in Tab.V

B. SARIMA implementation details

The seasonal ARIMA models are implemented using the SARIMAX function from the *statsmodels*[8] Python library. The SARIMAX function fits a model using the provided training data, and predicts a specified number of out-of-sample forecasts from the end of the training samples index. For the multivariate dataset, both historical load and historical records of the exogenous variables are presented to the model. The model orders for each time series are determined by analysing the ACF and PACF plots and by using the AIC criterion to select the optimal model. The model orders for all time series are reported in Table VI.

A single SARIMA model is fitted using the training dataset and, as the days in the test dataset are predicted, the previous actual observations for the test days and the validation data are made available for the model using the statsmodels `append` function. The `append` function stores the results for all training observations and extends the historical observations available to the model without refitting the model parameters.

REFERENCES

[1] T. Gneiting and M. Katzfuss, "Probabilistic forecasting," *Annual Review of Statistics and Its Application*, vol. 1, pp. 125–151, 2014.
[2] J. Lawless and M. Fredette, "Frequentist prediction intervals and predictive distributions," *Biometrika*, vol. 92, no. 3, pp. 529–542, 2005.

[7]https://keras.io/api/layers/recurrent_layers/lstm/
[8]www.statsmodels.org

TABLE V

OPTIMAL MODEL CONFIGURATIONS FOR NN-BASED MODELS FOR ALL DATASETS. THE ACRONYMS IN THE TABLE ARE: N_u: NUMBER OF UNITS IN LSTM LAYERS, N_h: NUMBER OF HIDDEN LAYERS, η: LEARNING RATE, λ_2: L2 REGULARIZATION PARAMETER, B: BATCH SIZE, d: DILATION FACTOR, N_f: NUMBER OF FILTERS IN CONVOLUTIONAL LAYERS IN EACH RESIDUAL BLOCK, k: SIZE OF CONVOLUTIONAL KERNEL, AND $\hat{q}_{lo}$ AND $\hat{q}_{hi}$ ARE THE LOWER AND UPPER QUANTILE LEVELS PREDICTED BY THE UNDERLYING NN IN ENCQR.

	TS	Model	LSTM					TCN						
			N_u	N_h	η	λ_2	$\hat{q}_{lo},\hat{q}_{hi}$	d	N_f	k	η	λ_2	B	$\hat{q}_{lo},\hat{q}_{hi}$
Portugal dataset	ID: 250	EnCQR	5	2	$1.0e^{-3}$	$5.0e^{-3}$	[0.05,0.92]	0	6	7	$7.0e^{-3}$	$5.0e^{-3}$	32	[0.20, 0.78]
		QRNN	57	2	$1.8e^{-3}$	$2.5e^{-3}$	-	3	18	6	0.025	$5.0e^{-3}$	64	-
		EnbPI	68	3	$1.0e^{-3}$	$5.0e^{-3}$	-	1	5	7	$5.0e^{-3}$	$5.0e^{-3}$	32	-
	ID: 77	EnCQR	48	1	$1.4e^{-4}$	$5.0e^{-3}$	[0.05,0.95]	0	6	7	$7.0e^{-3}$	$5.0e^{-3}$	32	[0.20, 0.78]
		QRNN	49	2	$2.5e^{-4}$	$1.8e^{-4}$	-	3	13	6	$2.6e^{-3}$	0.01	32	-
		EnbPI	59	3	$7.9e^{-4}$	$5.0e^{-3}$	-	1	8	7	$5.0e^{-3}$	$5.0e^{-3}$	32	-
	ID: 50	EnCQR	89	1	$1.0e^{-3}$	$5.0e^{-3}$	[0.09,0.89]	0	50	7	$7.0e^{-4}$	$5.0e^{-3}$	32	[0.19, 0.87]
		QRNN	93	2	$7.5e^{-4}$	$4.6e^{-4}$	-	4	8	7	$7.0e^{-4}$	$1.6e^{-3}$	32	-
		EnbPI	66	2	$2.0e^{-3}$	$5.0e^{-3}$	-	1	2	7	$2.0e^{-3}$	0.05	32	-
	ID: 90	EnCQR	69	2	$1.2e^{-3}$	$1.0e^{-3}$	[0.06,0.91]	1	50	7	$1.0e^{-3}$	$5.0e^{-3}$	32	[0.08, 0.92]
		QRNN	49	2	$2.5e^{-4}$	$1.8e^{-4}$	-	2	8	7	$7.0e^{-4}$	$1.6e^{-3}$	32	-
		EnbPI	99	2	$3.5e^{-3}$	$5.0e^{-3}$	-	1	9	7	$9.0e^{-3}$	0.05	32	-
	ID: 27	EnCQR	48	1	$1.4e^{-3}$	$5.0e^{-3}$	[0.14,0.94]	0	50	7	$1.0e^{-3}$	$5.0e^{-3}$	32	[0.1, 0.97]
		QRNN	166	3	$2.4e^{-4}$	0.058	-	0	23	7	$1.0e^{-3}$	$1.0e^{-3}$	32	-
		EnbPI	47	3	$1.0e^{-4}$	$5.0e^{-3}$	-	1	7	7	$8.0e^{-3}$	0.05	32	-
Elvia dataset	Industry	EnCQR	10	1	$1.0e^{-3}$	0.01	[0.10,0.80]	0	3	7	$2.5e^{-3}$	$5.0e^{-3}$	32	[0.02, 0.76]
		QRNN	100	2	$1.0e^{-4}$	$1.0e^{-3}$	-	2	2	5	0.01	0.05	32	-
		EnbPI	27	1	$5.0e^{-3}$	$5.0e^{-3}$	-	0	15	7	0.01	$5.0e^{-3}$	32	-
	Household	EnCQR	16	1	0.011	$5.0e^{-3}$	[0.17,0.79]	0	3	7	0.01	0.01	32	[0.10, 0.91]
		QRNN	122	2	$1.0e^{-4}$	$8.6e^{-4}$	-	0	3	4	0.01	$2.3e^{-3}$	56	-
		EnbPI	27	1	$5.0e^{-3}$	$5.0e^{-3}$	-	0	49	7	0.01	$5.0e^{-3}$	32	-
	Cabin	EnCQR	17	1	0.024	$5.0e^{-3}$	[0.12,0.85]	0	90	3	$6.0e^{-3}$	$1.0e^{-5}$	32	[0.04, 0.93]
		QRNN	138	3	$9.8e^{-4}$	0.029	-	1	2	5	$2.3e^{-3}$	$1.8e^{-4}$	64	-
		EnbPI	59	1	$9.0e^{-3}$	$5.0e^{-3}$	-	0	200	7	$9.0e^{-3}$	$5.0e^{-3}$	32	-
	Solar	EnCQR	89	1	$9.0e^{-4}$	$5.0e^{-3}$	[0.09,0.89]	1	101	7	$1.8e^{-3}$	$5.0e^{-3}$	32	[0.15, 0.99]
		QRNN	18	1	$5.0e^{-3}$	$5.0e^{-3}$	-	2	5	7	$3.5e^{-3}$	$5.0e^{-3}$	32	-
		EnbPI	147	2	$1.0e^{-3}$	$5.0e^{-3}$	-	2	5	7	$3.5e^{-3}$	$5.0e^{-3}$	32	-
	Wind	EnCQR	69	1	$9.0e^{-4}$	$5.0e^{-3}$	[0.05,0.99]	0	79	7	$5.0e^{-3}$	$5.0e^{-3}$	32	[0.05, 0.92]
		QRNN	18	3	$5.0e^{-3}$	$1.0e^{-3}$	-	1	12	7	$2.5e^{-3}$	$5.0e^{-3}$	32	-
		EnbPI	47	3	$1.0e^{-4}$	$5.0e^{-3}$	-	1	12	7	$2.5e^{-3}$	$5.0e^{-3}$	32	-
	Temperature	EnCQR	32	1	$5.0e^{-4}$	$1.0e^{-5}$	[0.15, 0.85]	4	32	7	$5.0e^{-4}$	$1.0e^{-5}$	32	[0.10, 0.91]
		QRNN	32	1	$5.0e^{-4}$	$1.0e^{-5}$	-	4	32	7	$2.5e^{-3}$	$1.0e^{-4}$	32	-
		EnbPI	32	2	$5.0e^{-4}$	$1.0e^{-4}$	-	4	64	7	$5.0e^{-4}$	$5.0e^{-4}$	32	-

TABLE VI

SEASONAL ARIMA MODEL PARAMETERS, (P,D,Q)×(P,D,Q)M. IN ELVIA, SOLAR, WIND, AND TEMPERATURE DATASETS EXOGENOUS VARIABLES ARE ADDED AND THE EXTENDED MODEL IS TERMED SARIMAX.

Portugal	
ID: 250	SARIMA (3,1,1)×(1,1,1)24
ID: 77	SARIMA (2,0,2)×(1,1,1)24
ID: 50	SARIMA (2,1,4)×(1,1,1)24
ID: 90	SARIMA (4,0,0)×(1,1,1)24
ID: 27	SARIMA (3,1,2)×(1,1,1)24
Elvia	
Industry	SARIMAX (1,0,2)×(1,1,1)24
Household	SARIMAX (4,1,1)×(1,1,1)24
Cabin	SARIMAX (2,1,1)×(1,1,2)24
Solar	
	SARIMAX (1,0,2)×(1,1,1)24
Wind	
	SARIMAX (4,0,1)×(0,0,0)24
Temperature	
	SARIMAX (2,1,1)×(1,1,1)24

[3] J. Nowotarski and R. Weron, "Recent advances in electricity price forecasting: A review of probabilistic forecasting," *Renewable and Sustainable Energy Reviews*, vol. 81, pp. 1548–1568, 2018.

[4] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi, and R. Jenssen, "Recurrent neural networks for short-term load forecasting: an overview and comparative analysis," *SpringerBriefs in Computer Science*, 2017.

[5] H. Liu, Z. Duan, and C. Chen, "A hybrid multi-resolution multi-objective ensemble model and its application for forecasting of daily pm2. 5 concentrations," *Information Sciences*, vol. 516, pp. 266–292, 2020.

[6] T. Evgeniou, M. Pontil, and A. Elisseeff, "Leave one out error, stability, and generalization of voting combinations of classifiers," *Machine learning*, vol. 55, no. 1, pp. 71–97, 2004.

[7] R. Foygel Barber, E. J. Candes, A. Ramdas, and R. J. Tibshirani, "The limits of distribution-free conditional predictive inference," *Information and Inference: A Journal of the IMA*, vol. 10, no. 2, pp. 455–482, 2021.

[8] C. Xu and Y. Xie, "Conformal prediction interval for dynamic time-series," in *International Conference on Machine Learning*. PMLR, 2021, pp. 11 559–11 569.

[9] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media, 2009.

[10] F. Huang, G. Xie, and R. Xiao, "Research on ensemble learning," in *2009 International Conference on Artificial Intelligence and Computational Intelligence*, vol. 3. IEEE, 2009, pp. 249–252.

[11] B. Kim, C. Xu, and R. Barber, "Predictive inference is free with the jackknife+-after-bootstrap," *Advances in Neural Information Processing Systems*, vol. 33, pp. 4138–4149, 2020.

[12] C. Zhang and Y. Ma, *Ensemble machine learning: methods and applications*. Springer, 2012.

[13] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, 1st ed. Chapman & Hall/CRC, 2012.

[14] G. Shafer and V. Vovk, "A tutorial on conformal prediction." *Journal of Machine Learning Research*, vol. 9, no. 3, 2008.

[15] Y. Romano, E. Patterson, and E. Candes, "Conformalized quantile regression," *Advances in neural information processing systems*, vol. 32, 2019.

[16] V. V. Gammerman A., Vovk V., "Learning by transduction," *Proceedings of the fourteenth conference on uncertainty in artificial intelligence*, pp. 48–155, 1998.

[17] C. Kath and F. Ziel, "Conformal prediction interval estimation and applications to day-ahead and intraday power markets," *International Journal of Forecasting*, vol. 37, no. 2, pp. 777–799, 2021.

[18] H. Papadopoulos, K. Proedrou, V. Vovk, and A. Gammerman, "Inductive confidence machines for regression," in *European Conference on Machine Learning*. Springer, 2002, pp. 345–356.

[19] M. Sesia and E. J. Candès, "A comparison of some conformal quantile regression methods," *Stat*, vol. 9, no. 1, p. e261, 2020.

[20] A. N. Angelopoulos and S. Bates, "A gentle introduction to conformal prediction and distribution-free uncertainty quantification," *arXiv preprint arXiv:2107.07511*, 2021.

[21] R. Koenker and G. Bassett Jr, "Regression quantiles," *Econometrica: journal of the Econometric Society*, pp. 33–50, 1978.

[22] J. W. Taylor, "A quantile regression neural network approach to estimating the conditional density of multiperiod returns," *Journal of Forecasting*, vol. 19, no. 4, pp. 299–311, 2000.

[23] Y. Wang, D. Gan, M. Sun, N. Zhang, Z. Lu, and C. Kang, "Probabilistic individual load forecasting using pinball loss guided lstm," *Applied Energy*, vol. 235, pp. 10–20, 2019.

[24] S. Smyl, "A hybrid method of exponential smoothing and recurrent neural networks for time series forecasting," *International Journal of Forecasting*, vol. 36, no. 1, pp. 75–85, 2020.

[25] G. Dudek, P. Pełka, and S. Smyl, "A hybrid residual dilated lstm and exponential smoothing model for midterm electric load forecasting," *IEEE Trans. Neural Netw. Learning Syst.*, vol. 33, no. 7, pp. 2879–2891, 2021.

[26] S. Smyl, G. Dudek, and P. Pelka, "ES-dRNN with dynamic attention for short-term load forecasting," in *2022 Int. J. Conf. Neural Netw. (IJCNN)*, 2022, pp. 1–8.

[27] D. Kivaranovic, K. D. Johnson, and H. Leeb, "Adaptive, distribution-free prediction intervals for deep networks," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 4346–4356.

[28] Y. Chen, Y. Kang, Y. Chen, and Z. Wang, "Probabilistic forecasting with temporal convolutional neural network," *Neurocomputing*, vol. 399, pp. 491 – 501, 2020.

[29] Y. Wang, D. Gan, M. Sun, N. Zhang, Z. Lu, and C. Kang, "Probabilistic individual load forecasting using pinball loss guided lstm," *Applied Energy*, vol. 235, pp. 10–20, 2019.

[30] M. Shepero, D. Van Der Meer, J. Munkhammar, and J. Widén, "Residential probabilistic load forecasting: A method using gaussian process designed for electric load data," *Applied Energy*, vol. 218, pp. 159–172, 2018.

[31] Y. Shen, X. Wang, and J. Chen, "Wind power forecasting using multi-objective evolutionary algorithms for wavelet neural network-optimized prediction intervals," *Applied Sciences*, vol. 8, no. 2, p. 185, 2018.

[32] A. Mashlakov, T. Kuronen, L. Lensu, A. Kaarna, and S. Honkapuro, "Assessing the performance of deep learning models for multivariate probabilistic energy forecasting," *Applied Energy*, vol. 285, p. 116405, 2021.

[33] A. Alexandrov, K. Benidis, M. Bohlke-Schneider, V. Flunkert, J. Gasthaus, T. Januschowski, D. C. Maddix, S. S. Rangapuram, D. Salinas, J. Schulz *et al.*, "Gluonts: Probabilistic and neural time series modeling in python." *J. Mach. Learn. Res.*, vol. 21, no. 116, pp. 1–6, 2020.

[34] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6405–6416.

[35] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: A system for large-scale machine learning," in *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, 2016, pp. 265–283.

[36] F. Chollet, "Keras," 2015.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *International Conference on Learning Representations, ICLR*, 2014.

[38] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, 2018.

[39] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *arXiv preprint arXiv:1312.6026*, 2013.