Supplementary Material - Adversarial Attacks on Probabilistic Autoregressive Forecasting Models

Raphaël Dang-Nhu [1] **Gagandeep Singh** [1] **Pavol Bielik** [1] **Martin Vechev** [1]

We provide the following three appendices:

- Appendix 1 provides proofs of Score-function Estimator and Reparametrization Estimator.
- Appendix 2 provides details of our datasets, preprocessings steps, architectures and hyper-parameters.
- Appendix 3 provides extended version of the experiments.

1. Proofs

Score-function Estimator. *In the general Bayesian setting where $\boldsymbol{y} \sim q[\cdot|\boldsymbol{x}+\boldsymbol{\delta}, z]$, the score-function gradient estimator of the expected value of $\chi(\boldsymbol{y})$ is:*

$$\nabla_{\boldsymbol{\delta}} \mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})] \\ \simeq \frac{\sum_{l=1}^{L} \chi(\boldsymbol{y}^l) q[z|\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{y}^l] \nabla_{\boldsymbol{\delta}} \log(q[\boldsymbol{y}^l|\boldsymbol{x}+\boldsymbol{\delta}, z])}{\sum_{l=1}^{L} q[z|\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{y}^l]}$$

where $\boldsymbol{y}^l$ is sampled from the prior distribution $q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}]$, and $q[z|\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{y}]$ denotes the probability that z is true knowing that $\boldsymbol{y}^l$ is generated.

Proof. The expectation is defined as the following integral over the output space:

$$\mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})] = \int_{\boldsymbol{y}} \chi(\boldsymbol{y}) q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z] dy$$

Under necessary regularity conditions (see following proof), we use Leibniz rule to obtain

$$\nabla_{\boldsymbol{\delta}} \mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})] = \int_{\boldsymbol{y}} \chi(\boldsymbol{y}) \nabla_{\boldsymbol{\delta}} q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z] dy$$

at every point $\boldsymbol{\delta}$ around which the gradient $\nabla_{\boldsymbol{\delta}} q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z]$ is locally continuous (in the model described in this paper, this regularity condition holds everywhere). The resulting integral can be transformed as follows into an expectation over the distribution $q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z]$.

$$\begin{aligned} &\nabla_{\boldsymbol{\delta}} \mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})] \\ &= \int_{\boldsymbol{y}} \chi(\boldsymbol{y}) \cdot \nabla_{\boldsymbol{\delta}} q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z] dy \\ &= \int_{\boldsymbol{y}} \chi(\boldsymbol{y}) q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z] \frac{\nabla_{\boldsymbol{\delta}} q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z]}{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z]} dy \\ &= \int_{\boldsymbol{y}} \chi(\boldsymbol{y}) q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z] \nabla_{\boldsymbol{\delta}} \log\left(q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z]\right) dy \\ &= \mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y}) \nabla_{\boldsymbol{\delta}} \log\left(q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z]\right)] \end{aligned}$$

This expectation can be approximated via Monte-Carlo methods. While it is in general not possible to directly sample from $q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}, z]$, what can be done instead is generating samples $\boldsymbol{y}^l$ for $l \in 1 \leq l \leq L$ from the prior $q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}]$, and attribute an importance weight to each of the resulting samples, yielding:

$$\nabla_{\boldsymbol{\delta}} \mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})] \\ \simeq \frac{\sum_{l=1}^{L} \chi(\boldsymbol{y}^l) q[z|\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{y}^l] \nabla_{\boldsymbol{\delta}} \log(q[\boldsymbol{y}^l|\boldsymbol{x}+\boldsymbol{\delta}, z])}{\sum_{l=1}^{L} q[z|\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{y}^l]}$$

The choice of $q[z|\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{y}^l]$ as the importance weight for $\boldsymbol{y}^l$ results from the application of Bayes rule:

$$q[\boldsymbol{y}^l|\boldsymbol{x}+\boldsymbol{\delta}, z] = \frac{q[z|\boldsymbol{x}+\boldsymbol{\delta}, \boldsymbol{y}^l]}{q[z|\boldsymbol{x}+\boldsymbol{\delta}]} q[\boldsymbol{y}^l|\boldsymbol{x}+\boldsymbol{\delta}]$$

□

Interversion of gradient and integral. *Suppose one of the following conditions is satisfied:*

1. *The model has a Gaussian likelihood and*

$$\chi(\boldsymbol{y}) = O(\exp(\|\boldsymbol{y}\|_1))$$

2. *The model has a Laplace likelihood and*

$$\chi(\boldsymbol{y}) = O(\exp(\sum_{i=t_0}^{T} \sqrt{|y_i|}))$$

[1] Department of Computer Science, ETH Zürich, Switzerland. Correspondence to: Raphaël Dang-Nhu <dangnhur@student.ethz.ch>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

3. *The model has a logistic likelihood and*

$$\chi(\boldsymbol{y}) = O(\exp(\sum_{i=t_0}^{T} \sqrt{|y_i|}))$$

Then the interchange of integration and differentiation for the score-function estimator is valid. In particular, all polynomially bounded statistics satisfy these conditions.

Following Theorem 2.4.3 in (Casella & Berger, 2002), let us denote

$$f(\boldsymbol{y}, \boldsymbol{\theta}) = \chi(\boldsymbol{y}) q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z].$$

Note that we call the perturbation $\boldsymbol{\theta}$ to have consistent notations, and that we consider $\boldsymbol{x}$ to be fixed as it does not change during the attack. f is differentiable and we have

$$\frac{\partial f}{\partial \boldsymbol{\theta}} = \chi(\boldsymbol{y}) \frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \boldsymbol{\theta}}. \tag{1}$$

In order to interchange integration and differentiation, Theorem 2.4.3 requires to dominate the rate of change

$$\left| \frac{f(\boldsymbol{y}, \boldsymbol{\theta}_0 + \boldsymbol{\delta}) - f(\boldsymbol{y}, \boldsymbol{\theta}_0)}{\boldsymbol{\delta}} \right|,$$

for $\|\boldsymbol{\delta}\|_1 \leq \boldsymbol{\delta}_0$, by an integrable function. In practice, the mean-value theorem yields

$$\left| \frac{f(\boldsymbol{y}, \boldsymbol{\theta}_0 + \boldsymbol{\delta}) - f(\boldsymbol{y}, \boldsymbol{\theta}_0)}{\boldsymbol{\delta}} \right| \leq \sup_{\boldsymbol{\epsilon} \in [\boldsymbol{0}, \boldsymbol{\delta}]} \left\| \frac{\partial f}{\partial \boldsymbol{\theta}} \Big|_{(\boldsymbol{y}, \boldsymbol{\theta}_0 + \boldsymbol{\epsilon})} \right\|_1,$$

and allows to instead bound the quantity

$$\sup_{\boldsymbol{\delta}, \|\boldsymbol{\delta}\|_1 \leq \boldsymbol{\delta}_0} \left\| \frac{\partial f}{\partial \boldsymbol{\theta}} \Big|_{(\boldsymbol{y}, \boldsymbol{\theta}_0 + \boldsymbol{\delta})} \right\|_1.$$

Equation (1) allows to express the partial derivative of f as a function of χ and q. Hence, we need to bound

$$\sup_{\boldsymbol{\delta}, \|\boldsymbol{\delta}\|_1 \leq \boldsymbol{\delta}_0} \left\| \chi(\boldsymbol{y}) \left. \frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \boldsymbol{\theta}} \right|_{(\boldsymbol{y}, \boldsymbol{\theta}_0 + \boldsymbol{\delta})} \right\|_1.$$

We define μ_i to be the mean predicted by the neural network for timestep i. Similarly, we define σ_i as the standard deviation predicted by the network. As i goes from t_0 to T, the chain rule yields

$$\frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \boldsymbol{\theta}} = \sum_{i=t_0}^{T} \frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \mu_i} \cdot \frac{\partial \mu_i}{\partial \boldsymbol{\theta}} + \frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \sigma_i} \cdot \frac{\partial \sigma_i}{\partial \boldsymbol{\theta}}. \tag{2}$$

Since μ_i and σ_i are learned by a neural network, their partial derivatives $\frac{\partial \mu_i}{\partial \boldsymbol{\theta}}$ and $\frac{\partial \sigma_i}{\partial \boldsymbol{\theta}}$ can be bounded by the global Lipschitz constant L of the network (it is not necessary to find the exact constant, an upper bound such as the one obtained in (Szegedy et al., 2013) is sufficient). Besides, let us denote $\psi_i(y_i, \mu_i, \sigma_i)$ the likelihood at timestep i. By definition, we have

$$q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z] = \prod_{j=t_0}^{T} \psi_j.$$

Since only ψ_i depends on μ_i and σ_i, we get

$$\frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \mu_i} = \frac{\partial \psi_i}{\partial \mu_i} \prod_{j \neq i} \psi_j,$$

and similarly

$$\frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \sigma_i} = \frac{\partial \psi_i}{\partial \sigma_i} \prod_{j \neq i} \psi_j.$$

Applied to equation (2), this yields

$$\left\| \frac{\partial q[\boldsymbol{y}|\boldsymbol{x} + \boldsymbol{\theta}, z]}{\partial \boldsymbol{\theta}} \right\|_1 \leq L \sum_{i=t_0}^{T} \left(\left\| \frac{\partial \psi_i}{\partial \mu_i} \prod_{j \neq i} \psi_j \right\|_1 + \left\| \frac{\partial \psi_i}{\partial \sigma_i} \prod_{j \neq i} \psi_j \right\|_1 \right). \tag{3}$$

Combined with equation (1), we obtain

$$\left\| \frac{\partial f}{\partial \boldsymbol{\theta}} \right\|_1 \leq |\chi(\boldsymbol{y})| L \sum_{i=t_0}^{T} \left(\left\| \frac{\partial \psi_i}{\partial \mu_i} \prod_{j \neq i} \psi_j \right\|_1 + \left\| \frac{\partial \psi_i}{\partial \sigma_i} \prod_{j \neq i} \psi_j \right\|_1 \right). \tag{4}$$

Here, we consider three cases for ψ: Gaussian, Laplace or logistic distribution.

Case 1 (Gaussian distribution)**.**

In the case of a Gaussian likelihood, we have

$$\psi_i(y_i, \mu_i, \sigma_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left[-\frac{1}{2} \left(\frac{y_i - \mu_i}{\sigma_i} \right)^2 \right],$$

After computations, we obtain

$$\frac{\partial \psi_i}{\partial \mu_i} = \frac{y_i - \mu_i}{\sigma_i} \cdot \psi_i = O\left(\exp\left(-|y_i|^{1.5}\right)\right)$$

and

$$\frac{\partial \psi_i}{\partial \sigma_i} = \left(\frac{(y_i - \mu_i)^2}{\sigma_i^3} - \frac{1}{\sigma_i} \right) \cdot \psi = O\left(\exp\left(-|y_i|^{1.5}\right)\right)$$

Besides, we have that

$$\prod_{j\neq i}\psi_j = O\left(\exp\left(-\sum_{j\neq i}|y_j|^{1.5}\right)\right).$$

Hence,

$$\left\|\frac{\partial\psi}{\partial\mu_i}\prod_{j\neq i}\psi_j\right\|_1 + \left\|\frac{\partial\psi}{\partial\sigma_i}\prod_{j\neq i}\psi_j\right\|_1 = O\left(\exp\left(-\sum_{i=t_0}^{T}|y_i|^{1.5}\right)\right). \tag{5}$$

Together with equation (4), this gives the following inequality

$$\begin{aligned}
&\left\|\frac{\partial f}{\partial\boldsymbol{\theta}}\right\|_1 \\
&\leq |\chi(\boldsymbol{y})|\cdot L\cdot\sum_{i=t_0}^{T}\left(\left\|\frac{\partial\psi}{\partial\mu_i}\prod_{j\neq i}\psi_j\right\|_1 + \left\|\frac{\partial\psi}{\partial\sigma_i}\prod_{j\neq i}\psi_j\right\|_1\right) \\
&= |\chi(\boldsymbol{y})|\cdot L\cdot O\left(\exp\left(-\sum_{i=t_0}^{T}|y_i|^{1.5}\right)\right).
\end{aligned}$$

Using the assumption that $\chi(\boldsymbol{y}) = O(\exp(||\boldsymbol{y}||_1))$,

$$\begin{aligned}
\left\|\frac{\partial f}{\partial\boldsymbol{\theta}}\right\|_1 &= O(\exp(||\boldsymbol{y}||_1))\cdot O\left(\exp\left(-\sum_{i=t_0}^{T}|y_i|^{1.5}\right)\right) \\
&= O\left(\exp\left(-\sum_{i=t_0}^{T}|y_i|(\sqrt{|y_i|}-1)\right)\right) \\
&= O(\exp(-||\boldsymbol{y}||_1))
\end{aligned}$$

All the asymptotic majorations are valid in the vicinity of $\boldsymbol{\theta}_0$, therefore we can take the sup on $\boldsymbol{\delta}$

$$\sup_{\boldsymbol{\delta},\|\boldsymbol{\delta}\|_1\leq\boldsymbol{\delta}_0}\left\|\left.\frac{\partial f}{\partial\boldsymbol{\theta}}\right|_{(\boldsymbol{y},\boldsymbol{\theta}_0+\boldsymbol{\delta})}\right\|_1 = O(\exp(-||\boldsymbol{y}||_1))$$

The right hand term is positive and integrable with respect to $\boldsymbol{y}$. This satisfies the domination condition of the theorem, and thus concludes the proof.

Case 2 (Laplace distribution)**.**

In the case of a Laplace distribution, we have

$$\psi_i(y_i,\mu_i,\sigma_i) = \frac{1}{2\sigma_i}\exp\left(-\left|\frac{y_i-\mu_i}{\sigma_i}\right|\right),$$

After computations, we obtain asymptotic majorations for the partial derivatives of ψ_i

$$\frac{\partial\psi_i}{\partial\mu_i} = -\frac{\operatorname{sign}(y_i-\mu_i)}{\sigma_i}\cdot\psi_i = O\left(\exp\left(-|y_i|^{0.75}\right)\right)$$

and

$$\frac{\partial\psi_i}{\partial\sigma_i} = \frac{|y_i-\mu_i|-1}{\sigma_i}\cdot\psi_i = O\left(\exp\left(-|y_i|^{0.75}\right)\right)$$

Besides,

$$\prod_{j\neq i}\psi_j = O\left(\exp\left(-\sum_{j\neq i}|y_j|^{0.75}\right)\right).$$

Using equation (4) (with a similar reasoning as for the Gaussian distribution), it follows that

$$\left\|\frac{\partial f}{\partial\boldsymbol{\theta}}\right\|_1 \leq |\chi(\boldsymbol{y})|\cdot L\cdot O\left(\exp\left(-\sum_{i=t_0}^{T}|y_i|^{0.75}\right)\right).$$

Again, using the assumption that

$$\chi(\boldsymbol{y}) = O\left(\exp\left(\sum_{i=t_0}^{T}\sqrt{|y_i|}\right)\right),$$

we get

$$\begin{aligned}
&\left\|\frac{\partial f}{\partial\boldsymbol{\theta}}\right\|_1 \\
&= O\left(\exp\left(\sum_{i=t_0}^{T}\sqrt{|y_i|}\right)\right)\cdot O\left(\exp\left(-\sum_{i=t_0}^{T}|y_i|^{0.75}\right)\right) \\
&= O\left(\exp\left(-\sum_{i=t_0}^{T}\sqrt{|y_i|}(|y_i|^{0.25}-1)\right)\right) \\
&= O\left(\exp\left(-\sum_{i=t_0}^{T}\sqrt{|y_i|}\right)\right).
\end{aligned}$$

The majoration being valid around $\boldsymbol{\theta}_0$, we also take the sup on $\boldsymbol{\delta}$

$$\sup_{\boldsymbol{\delta},\|\boldsymbol{\delta}\|_1\leq\boldsymbol{\delta}_0}\left\|\left.\frac{\partial f}{\partial\boldsymbol{\theta}}\right|_{(\boldsymbol{y},\boldsymbol{\theta}_0+\boldsymbol{\delta})}\right\|_1 = O\left(\exp\left(-\sum_{i=t_0}^{T}\sqrt{|y_i|}\right)\right).$$

The right-hand term is integrable and satisfies the domination condition of the theorem.

Case 3 (Logistic distribution)**.**

Finally, in the case of a logistic likelihood, we have

$$\psi_i(y_i,\mu_i,\sigma_i) = \frac{\exp\left(-\frac{y_i-\mu_i}{\sigma_i}\right)}{\sigma_i\left(1+\exp\left(-\frac{y_i-\mu_i}{\sigma_i}\right)\right)^2}.$$

Computations realized with a formal calculator yield

$$\frac{\partial\psi_i}{\partial\mu_i} = O\left(\exp\left(-|y_i|^{0.75}\right)\right)$$

and

$$\frac{\partial \psi_i}{\partial \sigma_i} = O\left(\exp\left(-|y_i|^{0.75}\right)\right)$$

We also have

$$\prod_{j\neq i} \psi_j = O\left(\exp\left(-\sum_{j\neq i}|y_j|^{0.75}\right)\right).$$

The rest of the proof is exactly similar to the case of a Laplace distribution.

Reparametrization Estimator. *Assume there exists a differentiable transformation $g_{\boldsymbol{x}}(\boldsymbol{\delta}, \boldsymbol{\eta})$ such that the random variable $\boldsymbol{y} \sim q[\cdot|\boldsymbol{x}+\boldsymbol{\delta}]$ can be reparametrized as $\boldsymbol{y} = g_{\boldsymbol{x}}(\boldsymbol{\delta}, \boldsymbol{\eta})$, where $\boldsymbol{\eta}$ is an independent random variable whose marginal distribution $p(\boldsymbol{\eta})$ is independent from $\boldsymbol{\delta}$. Then the importance sampling reparametrization estimator of the expectation's gradient is:*

$$\begin{aligned}&\nabla_{\boldsymbol{\delta}} \mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})]\\ \simeq&\nabla_{\boldsymbol{\delta}}\left(\frac{\sum_{l=1}^{L}\chi(g_{\boldsymbol{x}}(\boldsymbol{\delta},\boldsymbol{\eta}^l))q[z|\boldsymbol{x}+\boldsymbol{\delta},g_{\boldsymbol{x}}(\boldsymbol{\delta},\boldsymbol{\eta}^l)]}{\sum_{l=1}^{L}q[z|\boldsymbol{x}+\boldsymbol{\delta},g_{\boldsymbol{x}}(\boldsymbol{\delta},\boldsymbol{\eta}^l)]}\right)\end{aligned}$$

where for $1 \leq l \leq L$, $\boldsymbol{\eta}^l$ is sampled from the distribution $p(\boldsymbol{\eta})$, and $\boldsymbol{y}^l = g_{\boldsymbol{x}}(\boldsymbol{\delta}, \boldsymbol{\eta}^l)$.

Proof. Approximating the expectation $\mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})]$ via Monte-Carlo estimation with importance sampling yields:

$$\begin{aligned}&\nabla_{\boldsymbol{\delta}} \mathbb{E}_{q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta},z]}[\chi(\boldsymbol{y})]\\ \simeq&\nabla_{\boldsymbol{\delta}}\left(\frac{\sum_{l=1}^{L}\chi(\boldsymbol{y}^l)q[z|\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{y}^l]}{\sum_{l=1}^{L}q[z|\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{y}^l]}\right)\end{aligned}$$

where $\boldsymbol{y}^l$ is sampled from the prior distribution $q[\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{\delta}]$. With the assumptions of the theorem, we can rewrite:

$$\begin{aligned}&\left(\frac{\sum_{l=1}^{L}\chi(\boldsymbol{y}^l)q[z|\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{y}^l]}{\sum_{l=1}^{L}q[z|\boldsymbol{x}+\boldsymbol{\delta},\boldsymbol{y}^l]}\right)\\ =&\left(\frac{\sum_{l=1}^{L}\chi(g_{\boldsymbol{x}}(\boldsymbol{\delta},\boldsymbol{\eta}^l))q[z|\boldsymbol{x}+\boldsymbol{\delta},g_{\boldsymbol{x}}(\boldsymbol{\delta},\boldsymbol{\eta}^l)]}{\sum_{l=1}^{L}q[z|\boldsymbol{x}+\boldsymbol{\delta},g_{\boldsymbol{x}}(\boldsymbol{\delta},\boldsymbol{\eta}^l)]}\right)\end{aligned}$$

Since the respective effects of the perturbation and of randomness are decoupled in this final expression, it is differentiable with respect to $\boldsymbol{\delta}$, which concludes the proof. □

2. Experimental Details

Here we provide details of all our experiments to support reproducibility. Additionally, we will make all our datasets and source code available online.

2.1. Datasets and Preprocessing

S&P500 The S&P500 dataset is obtained via the *yfinance* API[1]. We focus on data-points between 1990/01 and 2000/12, identified by Fischer and Krauss (2018) as a period of exceptionally high trading returns compared to the following decades. We also follow Fischer and Krauss for preprocessing the data. A sequence of prices $\boldsymbol{p} = (p_1, \ldots, p_T)$ is first preprocessed to obtain a sequence of returns $(r_2, \ldots, r_T)$, defined as $r_i = \frac{p_i}{p_{i-1}} - 1$. Intuitively r_i is the gain (when positive) or loss obtained by investing one dollar in the stock at time $i-1$, and then selling at time i. Inversely, given a sequence of returns $\boldsymbol{r}$ and an initial price p_1, the corresponding sequence of prices can be obtained as:

$$p_k = p_1 \textstyle\prod_{i=2}^{k}(1 + r_i)$$

Both transformations are differentiable, which allows to perform the attack in the application space of prices rather than on returns. Besides, returns are normalized to have zero mean and unit variance. Denoting μ and σ for the mean and standard deviation of returns in the training set, the normalized sequence is $(\tilde{r}_2, \ldots, \tilde{r}_T)$, where $\tilde{r}_i = (r_i - \mu)/\sigma$. We refer to Fischer and Krauss (2018) for a thorough analysis of the properties of the S&P500 dataset.

Electricity Dataset We use the same preprocessing steps as described in (Salinas et al., 2019). Input sequences are divided by their average value v, and the corresponding prediction sequence is multiplied by v. This guarantees that all inputs are approximately in the same range.

2.2. Neural Architectures: S&P500 Dataset

LSTM The LSTM baseline used on the S&P500 dataset, we follow (Fischer & Krauss, 2018), and use a single LSTM layer with 25 hidden units, followed by a linear output layer. However, we use only one input neuron without activation instead of two neurons with softmax activation.

TCN In (Borovykh et al., 2017), several sets of hyperparameters for Temporal Convolutional Networks are used depending on the experiment. We decided to use 8 layers and a dilation of 2, in order to match as closely as possible the size of the LSTM receptive field. We selected the other parameters via grid-search, resulting in a kernel size of 2 and 3 channels. We use the TCN implementation provided by the authors of (Bai et al., 2018).

Ours For our probabilistic autoregressive model, we chose to use a single LSTM layer with 25 hidden units similar to the LSTM baseline, in order to guarantee the most fair comparison. We only changed the output layers to parametrize a Gaussian distribution. Following (Bishop,

[1] https://github.com/ranaroussi/yfinance

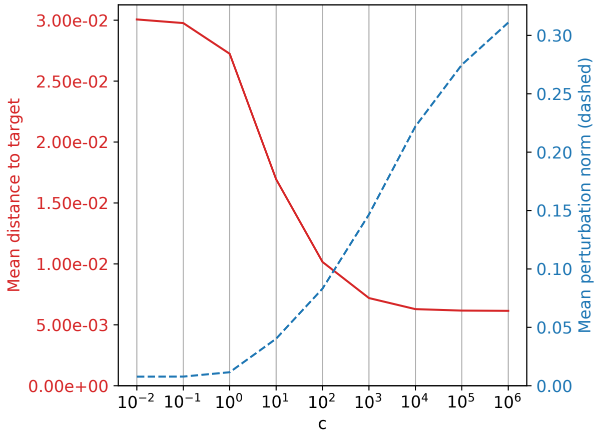

Figure 1. Perturbation norm and distance to target for different values of c when evaluated on the S&P500 Dataset.

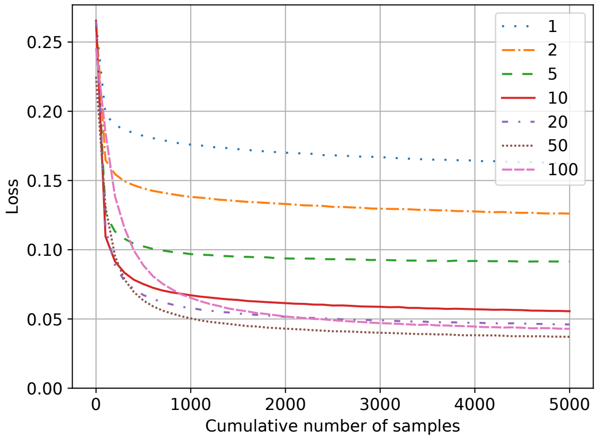

Figure 2. Attack loss for different values of L when evaluated on the S&P500 Dataset. The x-axis is the total number of generated samples rather than the number of perturbation updates, in order to provide a fair comparison in terms of attack computational cost.

1994), we use a linear layer without activation for the mean, and a linear layer with exponential activation for the scale of the distribution. We also performed experiments with a Gaussian mixture likelihood, but it did not improve the performance on our two benchmarks.

Training For both deterministic networks, we minimize mean-squared error on the training set. For our model, we use negative log-likelihood as a loss function. In both cases, we use the RMSPROP optimizer (Tieleman & Hinton, 2012) advised by Fischer and Krauss, with default parameters and learning rate of 0.01. We use an early-stopping patience of 20, and a large batch size of 2048 for training. Experiments with different values did not reveal a significant influence of these parameters.

2.3. Neural Architectures: Electricty Dataset

DeepAR The DeepAR architecture used for the Electricity experiments is based on a three-layer LSTM with 40 hidden units each. The number of samples used for Monte-Carlo estimation of the output is set to 200. The network is trained for 20 epochs with the Adam optimizer (Kingma & Ba, 2014), with batch size of 64 and learning rate of 0.001.

2.4. Attack Hyper-Parameters: S&P500 Dataset

For the S&P500 dataset, we optimize the attack objective function with the RMSPROP optimizer using a learning rate of 0.001 and 1000 iterations. These parameters were selected with a simple grid search because of the computational cost of running the attack repeatedly. The values used for the coefficient c are $10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3, 10^4, 10^5, 10^6$. We select the value that yields the best adversarial sample under the constraint that the perturbation norm is below the tolerance ϵ. The number of samples used to estimate the gradient is chosen to be $L = 50$.

Buy/Sell Attack We use $\lambda = 0.03$ for the target. For the Bayesian setting, we use $\gamma = y_{10}/x_{-1} = 1.0008$, in order to approximately balance the different classes. The 95% confidence interval is computed assuming Student's t-distribution. In the Bayesian case, the formula for the 95% confidence interval with importance sampling is derived in (Hesterberg, 1996).

Attack on Trading Strategies We use $\alpha = 0.1$ for the target scaling factor.

Influence of c In Figure 1, we examine the influence of tuning the attack objective function on average perturbation norm and distance to the attack target. We observe a trade-off between these two quantities that depends on the coefficient c: higher value for c yields better adversarial samples, at the cost of more input perturbation.

Influence of L We evaluate the effect of the number of samples L used in the reparametrization estimator on the attack loss in Figure 2. In this experiment, the value of c is fixed to 1000. We notice a trade-off in terms of convergence speed vs. final loss, that depends on the number of samples used for estimating the gradient. As a result, we choose to use $L = 50$ in our attacks.

2.5. Attack Hyper-Parameters: Electricity Dataset

We optimize the attack objective function with the ADAM optimizer. We use different optimizers for the two datasets so that the same optimizer is used for training the network

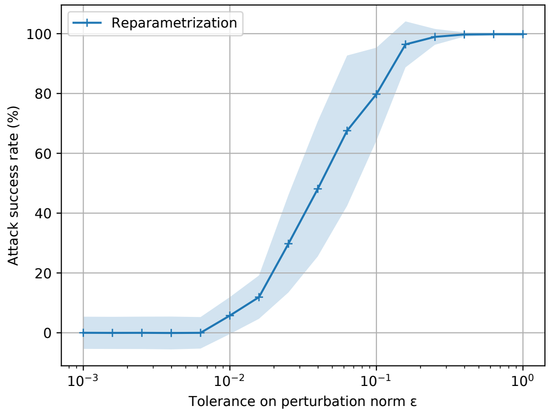

Figure 3. Success rate of the classification attack for different perturbation norms, in a Bayesian setting with observation $y_{10}/x_{-1} = \gamma = 1.0008$, and prediction horizon $h = 5$.

and to attack it. We use a learning rate of 0.01 and 1000 iterations. These parameters were also selected via informal search. The values used for the coefficient c are 0.1, 0.2, 0.3, 0.5, 0.7, 1, 2, 3, 5, 7, 10, 20, 30, 50, 70, 100, 200 and 300. The number of samples used to estimate the gradient is chosen to be $L = 50$.

3. Experimental Results

3.1. Trading Strategies

In Figure 4, we provide extended results for the long-short trading benchmark, with different horizons h and number of samples used for Monte-Carlo estimation of the prediction. We observe that the quality of the probabilistic prediction improves with the number of samples until 10^4 samples. Further increasing the number of samples does not yield significant performance improvements.

3.2. Evaluation of the Probabilistic Forecast

In Table 1, we give detailed results for the comparison of probabilistic forecasts quality with Ranked Probability Skill (a summary of these results is provided in Table 2). We observe that the forecasting quality of our model improves with the number of samples, and that an order of magnitude of the number of samples needed to obtain the best possible estimation is 10^4. As a comparison, the `DeepAR` implementation on the electricity dataset uses 200 samples. We surmise that this discrepancy is due to the low signal-to-noise ratio of financial data, that makes inference more difficult.

3.3. Bayesian Attack

In Figure 3, we plot the results of the classification attack in the Bayesian setting with observation $y_{10}/x_{-1} = \gamma$, where $\gamma = 1.0008$. We only implemented the reparametrization estimator, as the score-function estimator requires the overly complex estimation of $\nabla_{\boldsymbol{\delta}} \log(q[\boldsymbol{y}^l|\boldsymbol{x} + \boldsymbol{\delta}, z])$ for each sample $\boldsymbol{y}^l$. We observe that the attack success rate is very similar to the non-Bayesian setting, demonstrating that the reparametrization estimator adapts readily to the Bayesian setting. The attack success rate is approximately 80% for $\epsilon = 0.1$.

3.4. Electricity Dataset

In Figure 5, we show results of both over-estimation and under-estimation attacks on the electricity dataset, with examples of generated adversarial samples. We observe that for equal perturbation tolerance, the over-estimation attack yields mis-predictions of smaller amplitude. For instance, the reparametrization attack with $\epsilon = 0.8$ causes median over-estimation of around 15%, whereas it causes median under-estimation of 20%. We believe that this is due to the particular nature of the dataset rather than asymmetry in the attack. We do not observe such a discrepancy in the financial experiments.

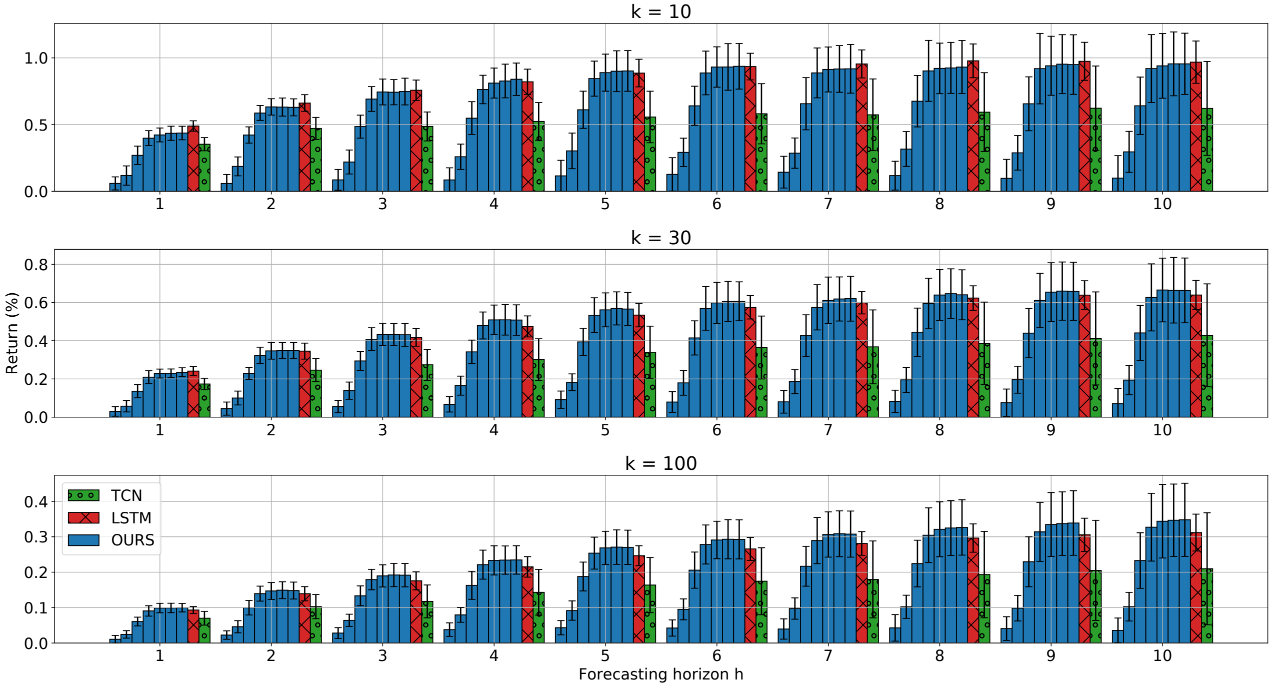

Figure 4. Financial gain on algorithmic trading tasks for different horizons h and portfolio sizes k (in $\%$ of the invested capital). The blue bars correspond to different number of samples for Monte-Carlo estimation of the prediction: from left to right $1, 10, 10^2, 10^3, 10^4, 10^5, 10^6$.

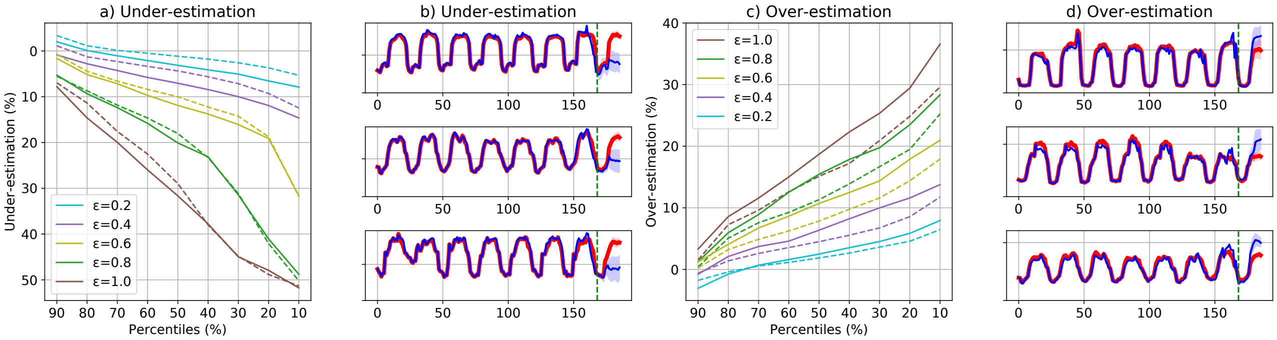

Figure 5. Results for the electricity dataset. a) Under-estimation of electricity consumption. For example, with $\epsilon = 1.0$, the attack using reparametrization estimator leads to under-estimation of at least 20% (y-axis) for 70% of samples (x-axis). b) Under-estimation adversarial samples. c) Over-estimation of electricity consumption. d) Over-estimation adversarial samples. In a) and b), results are given for reparametrization (continuous) and score-function (dashed) estimators. In c) and d), the reparametrization estimator is used, and ϵ is fixed to 0.9. Red curve is the original sample, blue curve is the generated adversarial sample. The vertical dashed line separates the input sequence from the network's prediction.

Table 1. Performance of different models on probabilistic forecast of various statistics. The comparison metric is Ranked Probability Skill (Weigel et al., 2007) of the prediction (lower scores correspond to better predictions). The performance of our architecture is given for different number of samples used in the Monte-Carlo estimation (1, 100, and 10000).

Statistics			Non-probabilistic		Probabilistic		
Name	h	π	`TCN` (Borovykh et al., 2017)	`LSTM` (Fischer & Krauss, 2018)	1 sample	Ours 100 samples	10000 samples
Cumulated Return	1	-	1.423 (± 0.022)	1.424 (± 0.016)	2.016 (± 0.023)	0.992 (± 0.002)	**0.982** (± **0.002**)
	5	-	1.468 (± 0.01)	1.466 (± 0.008)	1.992 (± 0.013)	1.0 (± 0.005)	**0.99** (± **0.004**)
	10	-	1.548 (± 0.029)	1.541 (± 0.019)	1.995 (± 0.011)	1.012 (± 0.008)	**1.002** (± **0.008**)
European Call Option	10	0.9	1.019 (± 0.004)	1.017 (± 0.003)	1.961 (± 0.22)	0.999 (± 0.009)	**0.989** (± **0.009**)
	10	1	1.122 (± 0.002)	1.121 (± 0.002)	1.966 (± 0.103)	0.992 (± 0.006)	**0.982** (± **0.005**)
	10	1.1	1.342 (± 0.002)	1.341 (± 0.002)	1.987 (± 0.03)	1.003 (± 0.007)	**0.993** (± **0.007**)
European Put Option	10	0.9	1.445 (± 0.021)	1.445 (± 0.017)	1.984 (± 0.015)	1.002 (± 0.007)	**0.992** (± **0.007**)
	10	1	1.302 (± 0.003)	1.3 (± 0.002)	1.957 (± 0.036)	0.984 (± 0.005)	**0.974** (± **0.005**)
	10	1.1	1.046 (± 0.005)	1.044 (± 0.002)	1.856 (± 0.094)	0.968 (± 0.004)	**0.959** (± **0.003**)
Limit Sell	10	1.01	2.822 (± 0.501)	3.137 (± 0.307)	1.917 (± 0.021)	1.013 (± 0.008)	**1.004** (± **0.008**)
	10	1.05	1.516 (± 0.001)	1.514 (± 0.002)	1.899 (± 0.027)	0.953 (± 0.006)	**0.944** (± **0.006**)
	10	1.20	1.035 (± 0.003)	1.034 (± 0.002)	1.792 (± 0.12)	0.951 (± 0.006)	**0.942** (± **0.005**)
Limit Buy	10	0.8	1.02 (± 0.002)	1.019 (± 0.002)	1.948 (± 0.223)	0.982 (± 0.015)	**0.972** (± **0.013**)
	10	0.95	1.412 (± 0.0)	1.41 (± 0.001)	1.926 (± 0.039)	0.967 (± 0.009)	**0.958** (± **0.008**)
	10	0.99	3.047 (± 0.012)	3.025 (± 0.028)	1.963 (± 0.024)	1.013 (± 0.006)	**1.003** (± **0.006**)

References

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

Bishop, C. M. Mixture density networks. 1994.

Borovykh, A., Bohte, S., and Oosterlee, C. W. Conditional time series forecasting with convolutional neural networks. *arXiv preprint arXiv:1703.04691*, 2017.

Casella, G. and Berger, R. L. *Statistical inference*, volume 2. Duxbury Pacific Grove, CA, 2002.

Fischer, T. and Krauss, C. Deep learning with long short-term memory networks for financial market predictions. *European Journal of Operational Research*, 270(2):654–669, 2018.

Hesterberg, T. C. Estimates and confidence intervals for importance sampling sensitivity analysis. *Mathematical and computer modelling*, 23(8-9):79–85, 1996.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization, 2014.

Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 2019.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.

Tieleman, T. and Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4 (2):26–31, 2012.

Weigel, A. P., Liniger, M. A., and Appenzeller, C. The discrete Brier and ranked probability skill scores. *Monthly Weather Review*, 135(1):118–124, 2007.