UNIVERSITY OF CALIFORNIA SAN DIEGO

Taming the Long Tail of Deep Probabilistic Forecasting

A thesis submitted in partial satisfaction of the
requirements for the degree Master of Science

in

Computer Science

by

Mayank Sharan

Committee in charge:

Professor Rose Yu, Chair
Professor Sicun Gao
Professor Jingbo Shang

2023

Copyright

Mayank Sharan, 2023

All rights reserved.

The thesis of Mayank Sharan is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to acknowledge Professor Rose Yu for her support as the chair of my committee. Through multiple drafts and many long nights, her guidance has proved to be invaluable.

I would like to acknowledge Jedrzej (Jacob) Kozerawski, without whom my research would have no doubt taken much longer. It is his support that helped me in an immeasureable way.

I would also like to acknowledge the exceptional researchers at Rose-STL lab for being an amazingly collaborative group improving each other's research.

I would like to thank my parents, Sneha Sharan and Tripurari Sharan, for their incredible support and my friends for bearing with me.

ABSTRACT OF THE THESIS

Taming the Long Tail of Deep Probabilistic Forecasting

by

Mayank Sharan

Master of Science in Computer Science

University of California San Diego, 2023

Professor Rose Yu, Chair

Deep probabilistic forecasting is gaining attention across numerous applications. From weather prognosis, electricity consumption estimation, traffic flow prediction, to autonomous vehicle trajectory prediction. However, existing approaches focus on improving on average metrics without addressing the long-tailed distribution of errors. This thesis identifies long tail behavior in the error distribution of state-of-the-art deep learning methods for probabilistic forecasting. We analyze potential sources and explanations for this behavior. Further, we present two loss augmentation methods to mitigate tailedness pf error distributions: Pareto Loss and Kurtosis Loss. Both methods modify the loss using the concept of moments to penalize higher loss samples. Kurtosis Loss uses a symmetric measure, the fourth moment,

while Pareto Loss uses an asymmetric measure of right-tailedness and models loss using a Generalized Pareto Distribution (GPD). We demonstrate the performance of our methods on several real-world datasets, including time series and spatiotemporal trajectories, achieving significant improvements on tail error metrics, while maintaining and even improving upon average error metrics.

Introduction

Probabilistic forecasting is one of the most fundamental problems in time series and spatiotemporal data analysis, with broad applications in energy, finance, and transportation. Deep learning models [Li et al., 2019, Salinas et al., 2020, Rasul et al., 2021a] have emerged as state-of-the-art approaches for forecasting rich time series and spatiotemporal data with uncertainty. In several forecast competitions, such as the M5 forecasting competition [Makridakis et al., 2020], Argoverse motion forecasting challenge [Chang et al., 2019], and IARAI Traffic4cast contest [Kreil et al., 2020], almost all the winning solutions are based on deep neural networks.

Despite encouraging progress, we observe that *the forecasting error for deep learning models has long-tail behavior*. This means that a significant amount of samples are very difficult to forecast. These samples have errors much larger than the average. Figure 1 visualizes an example of long-tail behavior for a motion forecasting task. Existing works often measure forecasting performance by averaging across test samples. However, average performance measured by metrics such as root mean square error (RMSE) or mean absolute error (MAE) can be misleading. A low RMSE or MAE may indicate good average performance, but it does not prevent the model from behaving disastrously in critical scenarios.

From a practical perspective, the long-tail behavior in forecasting error is alarming. In motion forecasting, the long tail could correspond to crucial events in driving, such as turning maneuver and sudden stops. Failure to accurately forecast in these scenarios would pose paramount safety risks in route planning. In electricity forecasting, these high errors could be during short circuits, power outages, grid failures, or sudden behavior changes. Focusing solely on average performance would ignore the electric load anomalies, significantly increasing

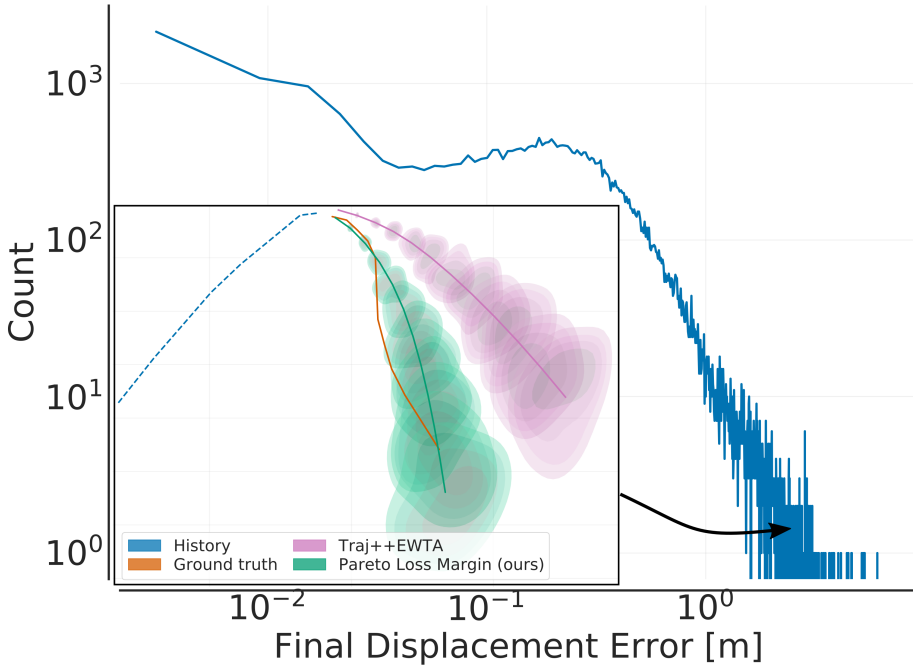

Figure 1. Log-log error distribution plot for trajectory prediction on the ETH-UCY dataset using SoTA (Traj++EWTA). We see the long tail in error upto 2 orders of magnitude higher than the average. Also shown is a tail sample with predictions from our method(teal) and Traj++EWTA(purple).

maintenance and operational costs.

Long-tailed learning is heavily studied in classification settings, with a focus on class imbalance. There is also rich literature for heavy-tailed time series [Kulik and Soulier, 2020]. However, long tail there usually refers to *distribution of the data*, not *distribution of the error*. We refer the reader to Table 2 in [Menon et al., 2020] and the survey paper [Zhang et al., 2021] for a complete review. Most common approaches to address the long-tail data distribution include post-hoc normalization [Pan et al., 2021], data resampling [Chawla et al., 2002, Torgo et al., 2013], loss engineering [Lin et al., 2017, Lu et al., 2018], and learning class-agnostic representations [Tiong et al., 2021]. These approaches implicitly assume strong correspondence between data and error. Hence, they are not directly applicable to forecasting, as we do not have pre-defined classes or the prediction error before training. [Makansi et al., 2021] observed similar long-tail error in trajectory and proposed to use Kalman filter prediction performance to measure sample

difficulty. However, Kalman filter is a different model class and its difficulties do not translate to deep neural networks used for forecasting.

In this paper, we address the long-tail behavior in prediction error for deep probabilistic forecasting. We present two loss augmentation methods: Pareto Loss and Kurtosis Loss. Kurtosis Loss is based on a symmetric measure of tailedness as a scaled fourth moment of a distribution. Pareto Loss uses the Generalized Pareto Distribution (GPD) to fit the long-tailed error distribution. The GPD can be described as a weighted summation of shifted moments, which is an asymmetric measure of tailedness. We investigate these measurements as loss regularization and reweighting approaches for probabilistic forecasting tasks. We achieve significantly improved tail performance compared to the base model and baselines. Interestingly, we also observe better average performance in most settings.

In summary, our contributions are

- We identify long-tail behavior in forecasting error for deep probabilistic models.
- We investigate principled approaches to address this long-tail behavior and propose two novel methods: Pareto Loss and Kurtosis Loss.
- We significantly improve the tail errors on four real world forecasting tasks, including two time series and two spatiotemporal trajectory forecasting datasets.

Chapter 1

Background

This chapter aims to provide some background and a gentle introduction to some key components of this work. Hopefully, this serves as a bridge of understanding for the uninitiated.

1.1 Forecasting models

Simply put, forecasting is making predictions for the future based on past and present data. Forecasting tasks are based on types of data from simple to complex. One of the simplest examples is predicting the next day of the week given the current day of the week. Since, this follows a strict set of rules it hardly seems like a prediction and more of a certainty. However, there are much more uncertain settings such as forecasting stock prices, the trajectory of a vehicle or weather.

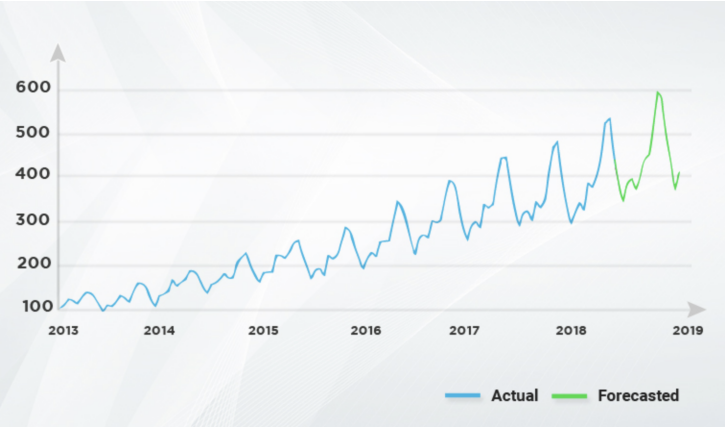

Figure 1.1. An illustration of what a forecast might look like

Figure 1.1 illustrates a sample forecast. Here the x axis is the different time steps at which the data is recorded and y axis is the value of the metric. The blue line is the behavior of the metric in the past and using that pattern we have made a prediction for the future timestemps shown as the green line. The data corresponding to the blue line is called the history and the data corresponding to the green line is called the forecast. The number of time steps for which a forecast is generated is known as the forecasting horizon. Forecasting is used for a wide range of applications. To name a few

- Economic forecasting such as inflation, GDP to guide monetary policy
- Energy forecasting to plan for generation and integrating renewable power
- Safety related such as earthquake prediction, flood forecasting
- Weather forecasting and meteorology for a wide range of usages

The class of methods tasked with producing these forecasts can be broadly categorized into qualitative and quantitative methods. Qualitative methods rely on subjective judgement and are often applicable in very specialized settings. Our focus here is to examine quantitative methods which generate forecasts as a function of past data and are known as forecasting models. These models can be as simple as moving averages to complex frameworks such as ARIMA or neural networks.

1.2 Probabilistic models

Events in the real world often have uncertainty associated with them. Models can be divided two categories based on the type of predictions they make. Point prediction models and probabilistic prediction models. Probabilistic models provide a lot more information than a point model does. They provide an expected prediction and a distribution for that prediction so that confidence intervals can be built and the user of the model has more information regarding

the range of outcomes to expect. On the other hand point models provide a single prediction outcome which is often the maximum likelihood prediction as per the model. For example, if your food delivery app tells you that your food is expected at a certain time that is a point prediction. However, if it gives you range from the earliest likely time of arrival to the latest likely time of arrival or, in a very mathematically inclined world, a gaussian distribution of the expected delivery time delivery, that would be a probabilistic prediction.

Probabilistic models are often more complex and require more data to learn than point models. Nevertheless, their nature often demands their usage in situations where guarantees and confidence intervals are needed. They are used for fault detection, traffic monitoring, robot control, speech recoognition, forecasting and much more.

1.3 Deep Probabilistic Forecasting models

Neural networks have been around since the 1940s. With improvements in compute power and numerous breakthroughs they have grown to find widespread application in pattern recognition and mathematical modeling. Over the last decade or so neural networks with multiple hidden layers or large number of parameters have established new performance benchmarks in a diverse set of machine tasks, often surpassing human performance. These models are called deep neural networks and the associated field is known as deep learning.

Forecasting models that use deep neural network architectures to generate probabilistic forecasts are known as deep probabilistic forecasting models. They are the state of the art in numerous forecasting settings. These models however do have certain limitations that we discover, analyze and mitigate in this work.

Chapter 2

Related Work

2.1 Deep probabilistic forecasting

There is a flurry of work on probabilistic forecasting using deep neural networks. A common practice is to combine classic time series models with deep learning, resulting in DeepAR [Salinas et al., 2020], Deep State Space [Rangapuram et al., 2018], Deep Factors [Wang et al., 2019] and normalizing Kalman Filter [de Bézenac et al., 2020]. Others introduce normalizing flow [Rasul et al., 2021b], denoising diffusion [Rasul et al., 2021a] and particle filter [Pal et al., 2021] to deep learning. For probabilistic trajectory forecasting, a few recent works propose to approximate the conditional distribution of future trajectories given the past with explicit parameterization [Tang and Salakhutdinov, 2019, Luo et al., 2020], CVAE [Sohn et al., 2015, Lee et al., 2017, Salzmann et al., 2020] or implicit models such as GAN [Gupta et al., 2018, Liu et al., 2019a]. Nevertheless, most existing works focus on average performance, the issue of long-tail in error distribution is largely overlooked in the community.

2.2 Long-tailed learning

The main efforts to address the long-tail in error in learning revolve around reweighing, resampling, loss function engineering, and two-stage training, but mainly for classification. Rebalancing during training is done in the form of synthetic minority oversampling [Chawla et al., 2002], oversampling with adversarial examples [Kozerawski et al., 2020], in-

verse class frequency balancing [Liu et al., 2019b], balancing using effective number of samples [Cui et al., 2019], or balance-oriented mixup augmentation [Xu et al., 2021].

Another direction involves post-processing in form of either normalized calibration [Pan et al., 2021] or logit adjustment [Menon et al., 2020]. An important direction is loss modification approaches such as Focal Loss [Lin et al., 2017], Shrinkage Loss [Lu et al., 2018], and Balanced Meta-Softmax [Ren et al., 2020]. Others use two-stage training [Liu et al., 2019b, Cao et al., 2019] or separate expert networks for the imbalance [Zhou et al., 2020, Li et al., 2020, Wang et al., 2021].

We refer the readers to [Zhang et al., 2021] for an extensive survey. [Tang et al., 2020] indicated SGD momentum can contribute to the aggravation of the long-tail problem and suggested de-confounded training to mitigate its effects. [Feldman, 2020, Feldman and Zhang, 2020] performed theoretical analysis and suggested label memorization in a long-tail distribution as a necessity for the network to generalize.

2.3 Imbalanced Regression

A few methods were developed for imbalanced regression. Many approaches are modifications of SMOTE (Synthetic Minority Oversampling Technique), such as, adapted to regression SMOTER [Torgo et al., 2013], augmented with Gaussian Noise SMOGN [Branco et al., 2017], or [Ribeiro and Moniz, 2020] extending for the prediction of extremely rare values.

[Steininger et al., 2021] proposed DenseWeight, a method based on Kernel Density Estimation for better assessment of the relevance function for sample reweighing. [Yang et al., 2021] proposed a distribution smoothing over label (LDS) and feature space (FDS) for imbalanced regression. [Prasad et al., 2018, Zhu and Zhou, 2021] worked on robust regression approaches applicable to point forecast. GARCH [Bollerslev, 1986] and AFTER [Cheng et al., 2015] addressed heavy-tailed error in forecasting but both are statistical models, and not applicable to deep learning.

A concurrent work is [Makansi et al., 2021] where they also notice the long-tail error distribution for trajectory prediction. They use Kalman filter [Kalman, 1960] performance as a difficulty measure and propose contrastive learning to mitigate the tail problem. However, the tail samples of Kalman Filter differ from that of deep learning models.

Most methods in long-tailed learning operate on *known heavy-tailedness* in data, whereas our focus is to mitigate the unknown long tail in the error distribution of test samples without any specific assumption on the data distribution. This is essential to our problem setting and techniques.

Chapter 3

Methodology

We first identify the long-tail error distribution in probabilistic forecasting. Then, we propose two novel methods, Pareto Loss and Kurtosis Loss, to mitigate the long tail in error.

3.1 Long-tail in probabilistic forecasting

Given input $x_t \in \mathbb{R}^{d_{in}}$ and output $y_t \in \mathbb{R}^{d_{out}}$ respectively, probabilistic forecasting task aims to predict the conditional distribution of future states $\mathbf{y} = (y_{t+1}, \ldots, y_{t+h})$ given current and past observations $\mathbf{x} = (x_{t-k}, \ldots, x_t)$ as:

$$p(y_{t+1}, \ldots, y_{t+h} | x_{t-k}, \ldots, , x_t) \tag{3.1}$$

where k is the length of the history and h is the prediction horizon. The maximum likelihood prediction –mean when the predicted distribution is a Gaussian– can be denoted as $\hat{\mathbf{y}} = (\hat{y}_{t+1}, \ldots, \hat{y}_{t+h})$.

Long tailed error distributions for deep learning models manifest in numerous real world datasets. This is evident in four benchmark forecasting datasets studied in this work (Time series: Electricity [Dua and Graff, 2017], Traffic [Dua and Graff, 2017]; Trajectory: ETH-UCY [Pellegrini et al., 2009, Lerner et al., 2007], nuScenes [Caesar et al., 2020]). Figure 3.1 shows the long-tailed error distribution for time series datasets for DeepAR [Salinas et al., 2020]

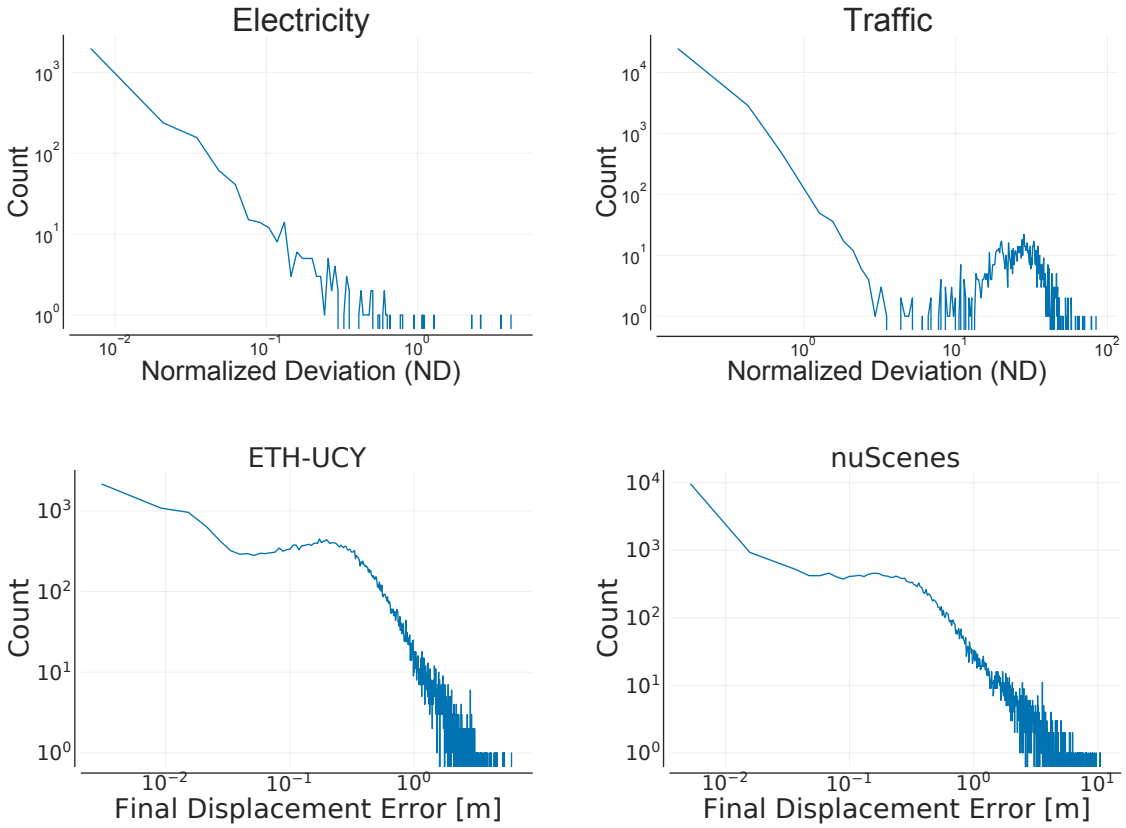

Figure 3.1. Log-log error distribution plots. Time series datasets (upper half) use DeepAR, trajectory datasets (bottom half) use Traj++EWTA. This clearly illustrates the long tail in error distribution.

and for trajectory datasets using Trajectron++EWTA [Makansi et al., 2019]. We follow the literature and use Normalized deviation (ND) and Final Displacement Error (FDE) to measure the performance. Log scale is used to improve the visibility of the tail.

We also observe that the samples forming the tail in error vary across methods and even across different runs of the same model. For example, we trained 2 DeepAR [Salinas et al., 2020] models on the same Electricity forecasting dataset from UCI repository [Dua and Graff, 2017]. We observe that the sets of samples with the top 5% error values have only 3.5% samples common to both models. This shows that the tail in the data does not necessarily correspond to the tail in error. We do further experiments to validate this hypothesis. They are discussed in

Section 4.2.2 and Section 5.1.

The fact that it is impossible to identify a fixed set of tail samples means that we cannot simply reweigh ([Cui et al., 2019, Fan et al., 2017]) or resample ([Torgo et al., 2013, Branco et al., 2017]) these samples before training. The variation of tail samples between models also invalidates the approach taken by [Makansi et al., 2021]. Mitigating the long tail in error requires an approach that is independent of the data distribution and is adaptive during training. Thus, we propose using tail-sensitive loss augmentations that adapt the model to also improve on samples with tail errors.

3.2 Pareto Loss

Long tail distributions naturally lend themselves to analysis using Extreme Value Theory (EVT). EVT [McNeil, 1997] shows that long tail behavior of a distribution can be modeled as a generalized Pareto distribution (GPD). The probability distribution function (pdf) of the GPD is:

$$f_{(\xi,\eta,\mu)}(a) = \frac{1}{\eta}\left(1+\xi\left(\frac{a-\mu}{\eta}\right)\right)^{-(\frac{1}{\xi}+1)} \Rightarrow \quad f_{(\xi,\eta)}(a) = \left(1+\frac{\xi a}{\eta}\right)^{-(\frac{1}{\xi}+1)} \tag{3.2}$$

where the parameters are location (μ), scale (η) and shape (ξ). Without loss of generality, μ can be set to 0. We can drop the scaling term $\frac{1}{\eta}$ as the pdf will be scaled using a hyperparameter.

The idea behind our Pareto Loss is to fit the GPD pdf in (3.2) to the final loss distribution and use it to increase the emphasis placed on the tail samples during training. We denote the loss function of a given model, base loss, as l. In probabilistic forecasting, a commonly used loss is Negative Log Likelihood (NLL) loss: $l_i = -\log(p(\mathbf{y}^{(i)}|\mathbf{x}^{(i)}))$ where $\langle \mathbf{x}^{(i)}, \mathbf{y}^{(i)} \rangle$ is the i^{th} training sample.

Our goal is to reduce the long-tail error measured by, e.g. MSE. This means that using NLL to fit the GPD might not lead to the intended prioritization of samples. Thus, we propose using an auxiliary loss $\hat{l}$, which is better correlated with the evaluation metric used, to fit the

GPD. The choice of auxiliary loss is completely up to the model designer and could be the base loss itself in settings where it correlates well with the evaluation metric. See Appendix F for further details.

There are two main classes of loss augmentation methods to mitigate tail errors: *regularization* [Ren et al., 2020, Makansi et al., 2021] and *reweighting* [Lin et al., 2017, Lu et al., 2018, Yang et al., 2021]. Inspired by these, we propose two variations of the Pareto Loss using the GPD fitted on $\hat{l}$: Pareto Loss Margin (PLM) and Pareto Loss Weighted (PLW).

PLM is based on margin-based regularization [Ren et al., 2020, Liu et al., 2016], which assigns larger additive penalties to tail samples using the fitted GPD. For a given hyperparameter λ, PLM is defined as,

$$l_{plm} = l + \lambda * r_{plm}(\hat{l}), \quad r_{plm}(\hat{l}) = 1 - f_{(\xi,\eta)}(\hat{l}) \tag{3.3}$$

An alternative is to reweigh the loss terms using the fitted GPD. For a given λ, PLW is defined as,

$$l_{plw} = w_{plw}(\hat{l}) * l, \quad w_{plw}(\hat{l}) = 1 - \lambda * f_{(\xi,\eta)}(\hat{l}) \tag{3.4}$$

3.3 Kurtosis Loss

Use cases requiring higher emphasis on the extreme tail need an even more skewed measure of heavy-tailedness. For such cases we propose using Kurtosis, which is the scaled fourth moment relative to its mean. It assesses the propensity of a distribution to have extreme values within its tails. To increase the emphasis on tail samples, we use this measure as a margin-based regularization term in our proposed Kurtosis Loss. For a given hyperparameter λ and using the same notations as Sec.3.2, Kurtosis Loss is defined as,

$$l_{kurt} = l + \lambda * r_{kurt}(\hat{l}), \quad r_{kurt}(\hat{l}) = \left(\frac{\hat{l} - \mu_{\hat{l}}}{\sigma_{\hat{l}}} \right)^4 \tag{3.5}$$

where $\mu_{\hat{l}}$ and $\sigma_{\hat{l}}$ are the mean and standard deviation of the auxiliary loss ($\hat{l}$) for a batch of samples.

We do not use a reweighting based approach with kurtosis as there is no upper bound to the kurtosis value. This could lead to convergence issues due to very high weights for some samples.

3.4 Connection between Pareto and Kurtosis Loss

Kurtosis Loss and Pareto Loss are both based on moments of a distribution. Pareto Loss is a weighted sum of shifted moments, while Kurtosis Loss is the scaled fourth moment. Specifically, let $b = \frac{\xi a}{\eta}$ and $c = -(\frac{1}{\xi} + 1)$, then the Taylor expansion for the GPD pdf in (3.2) is,

$$(1+b)^c = 1 + cb + \frac{c(c-1)}{2!}b^2 + \frac{c(c-1)(c-2)}{3!}b^3 + \cdots \tag{3.6}$$

For $c < 0$ or equivalently $\xi < -1$ or $\xi > 0$, the coefficients are positive for even moments and negative for odd moments (odd and even powers of b). Even moments are always symmetric and positive, whereas odd moments are positive only for right-tailed distributions. Since we use the negative of the pdf, it yields an asymmetric measure of the right-tailedness of the distribution.

Kurtosis Loss uses the fourth moment. This is a symmetric and positive measure. GPD and kurtosis are visualized in Appendix E. Kurtosis emphasizes extreme values in the tail. Our experiments also show that it is more effective in controlling the extremes in the error distribution.

Chapter 4

Experiments

We evaluate our methods on multiple benchmark datasets from two probabilistic forecasting tasks: time series forecasting (1D) and trajectory prediction (2D).

4.1 Setup

4.1.1 Datasets

For time series forecasting, we use electricity and traffic datasets from the UCI ML repository [Dua and Graff, 2017] used in [Salinas et al., 2020] as benchmarks. We also generate numerous synthetic 1D time series datasets to further our understanding of the potential causes of long tail distribution. We generate 4 different sets of datasets which will be described in the following sections.

For trajectory prediction, we use two benchmark datasets: a pedestrian trajectory dataset ETH-UCY (which is a combination of ETH [Pellegrini et al., 2009] and UCY [Lerner et al., 2007] datasets) and a vehicle trajectory dataset nuScenes [Caesar et al., 2020]. Further details regarding the datasets are available in Appendix A.

4.1.2 Baselines

We compare with SoTA baselines in long tail mitigation for different tasks:

- Contrastive Loss: [Makansi et al., 2021] uses contrastive loss as a regularizer to group

examples together. The grouping is based on Kalman Filter prediction errors as a measure of sample difficulty.

- Label Distribution Smoothing (LDS): [Yang et al., 2021] uses a symmetric kernel to smooth the label distribution and use its inverse to reweigh the loss terms.
- Shrinkage Loss: [Lu et al., 2018] uses a sigmoid-based function to reweigh loss terms. This deprioritizes lower loss values.
- Focal Loss: [Lin et al., 2017] uses L1 loss to reweigh the loss terms. Additional power of the loss term increases the steepness of the loss function.

Focal Loss, Shrinkage Loss, and LDS were originally proposed for classification and/or regression and required adaptation to be applied to the forecasting task. See Appendix B for details.

4.1.3 Evaluation Metrics

We use metrics in accordance with literature [Walters et al., 2021, Salzmann et al., 2020, Makansi et al., 2021]: Average Displacement Error (ADE), which is the average L2 distance between total predicted trajectory and ground truth, and Final Displacement Error (FDE) which is the L2 distance for the final timestep. For time series forecasting, we use the metrics from DeepAR [Salinas et al., 2020] and use Normalized Deviation (ND) and Normalized Root Mean Squared Error (NRMSE). We also report Continuous Ranked Probability Score (CRPS) [Gneiting and Ranjan, 2011] for the time series datasets, a more suitable metric for probabilistic forecasting.

Apart from the above-mentioned average performance metrics, we introduce metrics to capture the tail errors. We adapt the Value-at-Risk (VaR (4.1)) tail metric from financial domain:

$$\mathrm{VaR}_\alpha(E) = \inf\{e \in E : P(E \geq e) \leq 1 - \alpha\} \tag{4.1}$$

VaR at level $\alpha \in (0,1)$ is the smallest error e such that the probability of observing error greater than e is less than $1-\alpha$, where E is the error distribution. This evaluates to the α^{th} quantile of the error distribution. We measure VaR at three different levels: 0.95, 0.98, and 0.99. Additionally, we report the maximum error representing the worst-case performance. We present tail metrics on the complete error distribution as there is no fixed set of tail samples across different methods (See Section 3.1).

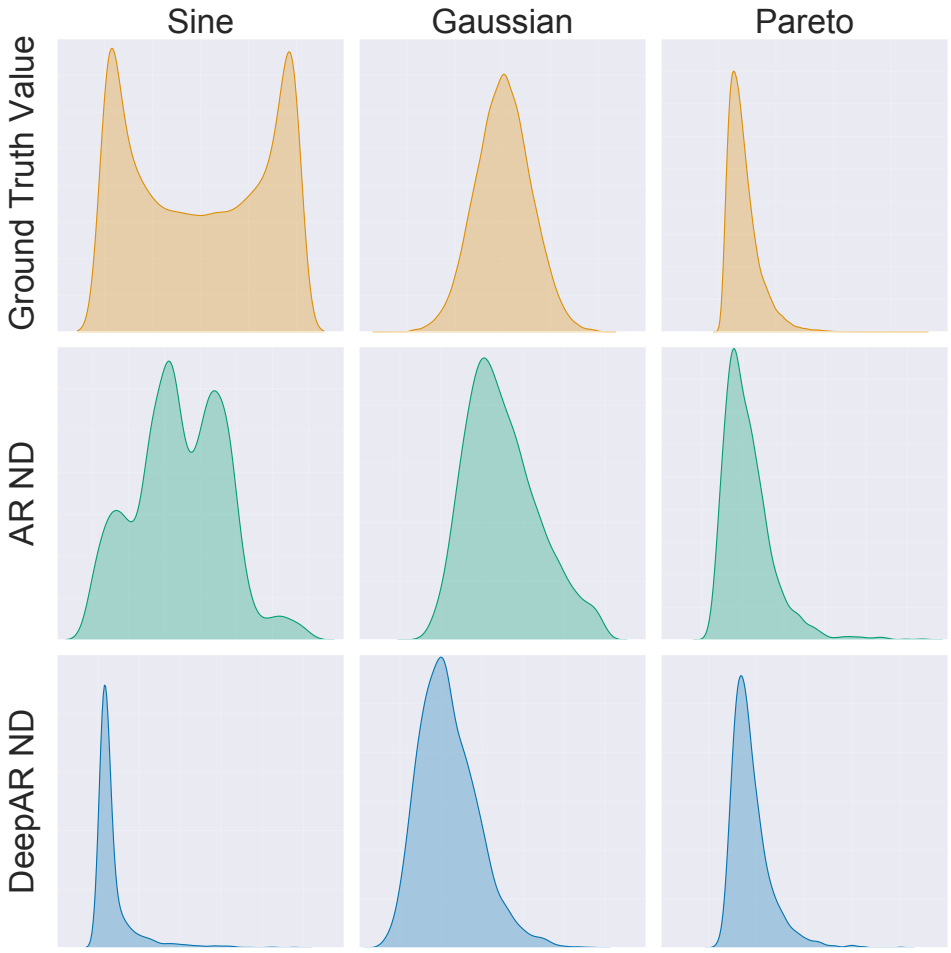

Figure 4.1. Top Row: Ground truth distribution for synthetic datasets. Middle Row: ND error distribution using AR. Bottom Row : ND error distribution using DeepAR. Datasets (L to R): Sine, Gaussian, Pareto. **Note**: the x-axes for plots in the same column or y-axes for plots in the same row are not for the same range of values.

4.2 Synthetic Dataset Experiments

4.2.1 Different distribution datasets

To understand the impact of long tail data on long tail in error, we perform experiments on three synthetic datasets. These datasets are generated using different base distributions to get different degrees of tail in data. We generate Sine, Gaussian and Pareto datasets. The task is to forecast 8 steps ahead given a history of 8 time steps. We use AutoRegression (AR) and DeepAR [Salinas et al., 2020] as models to perform this task. The top row in Figure 4.1 shows that among the datasets, only Gaussian and Pareto exhibit tail in the data distribution. The data distribution is available here because the datasets were generated synthetically.

On the Sine dataset, we observe long tail error for DeepAR but not for AR. This is especially significant as there is *no long tail in the data distribution*. On Gaussian and Pareto datasets, DeepAR leads to a heavier tail than AR, suggesting that the long tail in data also contributes to long tail in error. The difference between AR and DeepAR error distributions also invalidates the assumption made by [Makansi et al., 2021]. Using the prediction performance from Kalman Filter is not a good indicator of sample tailedness for deep neural networks. The complete results for these datasets are available in appendix L.

4.2.2 1-step Forecasting

The results we get in Section 4.2.1 while interesting are not conclusive. There are numerous factors that can lead to the outcomes, which we do not control. Such as, since we are forecasting for 8 time steps, certain sequences may be more unlikely which may not be reflected in a simple data distribution. So for clearer insights we designed 1-step forecasting experiments. The task is to forecast 1 step ahead given a history of 7 time steps. This design takes away confounding factors and allows us to examine the exact prediction distribution vs the error.

Another factor that complicates our previous experiment is the complexity of the history. We do not control how complex or varied the history data in each time series is. This could also

impact whether the model is able to predict well. So we generated 3 sets of datasets with different history complexities. The first set (same-history) has identical sinusoidal history for all series in the dataset. This is a sanity check dataset as for the same input, the model would predict the same value and we would see the tail in error directly correspond to the tail in data. The second set (different-sine) has different sinusoidal histories for each series in the dataset. Sinusoid is a fixed sequential function and thus relatively easier to model. The third set (different-gaussian) has different autoregressive histories for each series in the dataset generated using Gaussian sampling. These histories are a bit more difficult to model than a sinusoidal series as they are generated using sampling.

The final factor is the tailedness of the data we need to predict. We use different distributions to simulate different severities of tail in data in Section 4.2.1. However, it is better to use the same distribution and have different tail lengths to ensure that similar structure in data is maintained. To this end, we generate 5 different datasets in each set (same-history, different-sine, and different-gaussian) using different generalized pareto distributions to sample the point to be predicted. These pareto distributions have different shape parameters and scaling ratios to generate data with different severities of tail. The percentile measures for each of these distributions is shown in Table 4.1 which illustrates the difference in severity of tail in data. The shape of these distributions is shown in Figure 4.2.

Table 4.1. Data statistics for the different pareto distributions used to generate the point to predict for 1-step forecasting datasets. The median stays similar across all distributions however the percentiles above 90 increase significantly. This is due to the longer tail of the distribution.

DISTRIBUTION NAME	SHAPE	RATIO	MEDIAN	90^{th} %TILE	95^{th} %TILE	99^{th} %TILE
PARETO_0.1_3	0.1	3	0.64	1.39	1.60	1.91
PARETO_0.1_2	0.1	2	0.70	1.97	2.33	2.87
PARETO_0.1_1	0.1	1	0.70	2.47	3.33	5.74
PARETO_0.3_1	0.3	1	0.75	3.13	4.57	9.65
PARETO_0.5_1	0.5	1	0.80	4.02	6.43	17.3

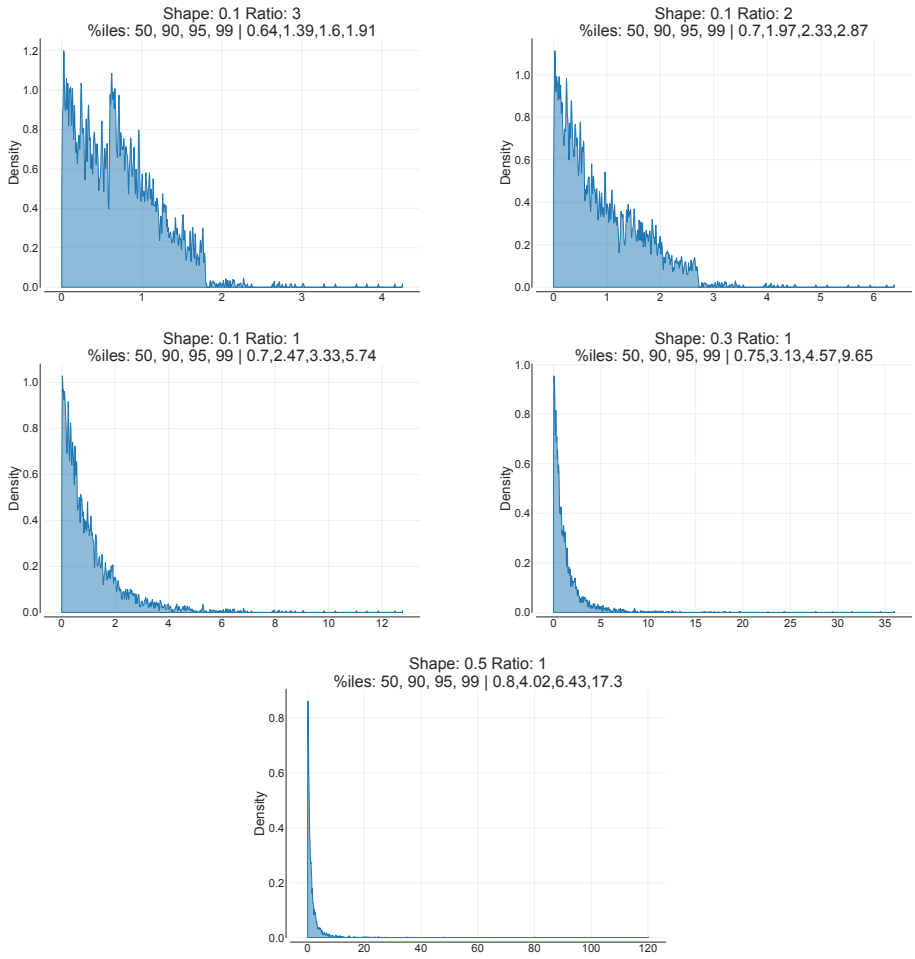

Figure 4.2. Data distribution for the prediction point generated using different generalized pareto distributions. Top Row: Pareto_0.1_3, Pareto_0.1_2; Middle Row: Pareto_0.1_1, Pareto_0.3_1; Bottom Row: Pareto_0.5_1. The increasing severity of the tail can be seen in these plots.

Details of the generation process for these datasets can be found in Appendix G. We use DeepAR [Salinas et al., 2020] as the model to perform the forecasting tasks in this experiment for consistency with other experiments. We examine the error distribution across the different factors of history complexity and prediction data tail length to gain a better understanding of the phenomenon. The results for these experiments are presented and discussed in Section 5.1.

4.3 Real-World Experiments

4.3.1 Time Series Forecasting

We present average and tail metrics using ND and NRMSE for the time series forecasting task on electricity and traffic datasets in Tables 5.2 and 5.3 respectively. All methods use DeepAR [Salinas et al., 2020], one of the SoTA in probabilistic time series forecasting, as the base model. The task for both datasets is to use a 1-week history (168 hours) to forecast for 1 day (24 hours) at an hourly frequency. The base model exhibits long tail behavior in error on both datasets (see Figure 3.1). The tail of the error distribution is significantly longer for the traffic dataset as compared to the electricity dataset. This is evident from comparing the tail error values to the average error. The auxiliary loss used here is MAE to correlate with L1 metrics like ND. DeepAR can have intrinsic variation on re-training so results in Table 5.2 are averaged over 3 runs.

4.3.2 Trajectory Forecasting

We present experimental results on ETH-UCY and nuScenes datasets in Tables 5.4 and 5.5 respectively. Following [Salzmann et al., 2020] and [Makansi et al., 2021] we calculate model performance based on the best out of 20 guesses. On both datasets, we compare with several long-tail baselines using Trajectron++EWTA [Makansi et al., 2021] as a base model due to its SoTA average performance on these datasets. The auxiliary loss used here is MAE with MSE to correlate with L2 metrics like ADE and FDE.

Chapter 5

Results and Discussion

5.1 1-step Forecasting Experiments

For each of the 1-step forecasting datasets we analyze ND error and the Spearman Correlation coefficient between the error and the pdf of the prediction point data. The intent is to examine the effect the long tail of data has on the long tail of error. We choose to analyze this with ND error as it is linear with respect to the prediction label making the analysis clearer.

5.1.1 Same-History Datasets

These datasets are designed to establish a baseline and a sanity check for long tail behavior. Since the history is identical for all series the predictions made by the model will be as well. This leads to linearly increasing error on either side of the prediction. The prediction label vs ND error plots (Figure 5.1) show this.

5.1.2 Different-Sine Datasets

These datasets are designed to compare tail errors across distributions given simple sinusoidal histories. As expected, the prediction by the model is not identical for different samples. This leads to variation in error for similar prediction labels. The relation between label and error is not as simple as for the Same-History datasets (Section 5.1.1) but roughly follows the same pattern. The prediction label vs ND error plots are available in Figure 5.2.

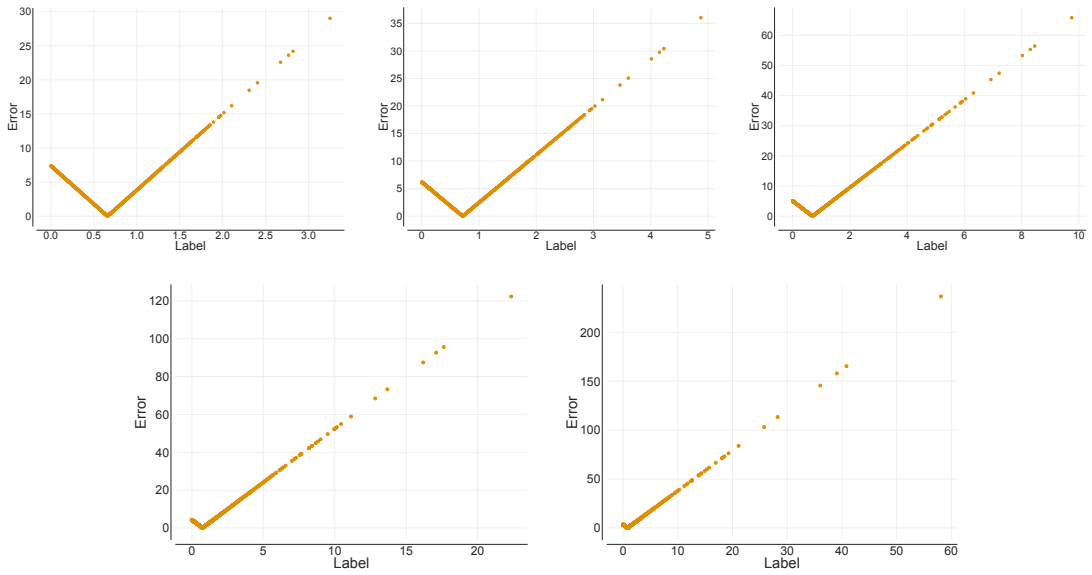

Figure 5.1. Prediction label vs ND error for the same history datasets. Top Row: Pareto_0.1_3, Pareto_0.1_2, Pareto_0.1_1; Bottom Row: Pareto_0.3_1, Pareto_0.5_1. We can see that the error directly corresponds to the label as the model makes identical predictions.

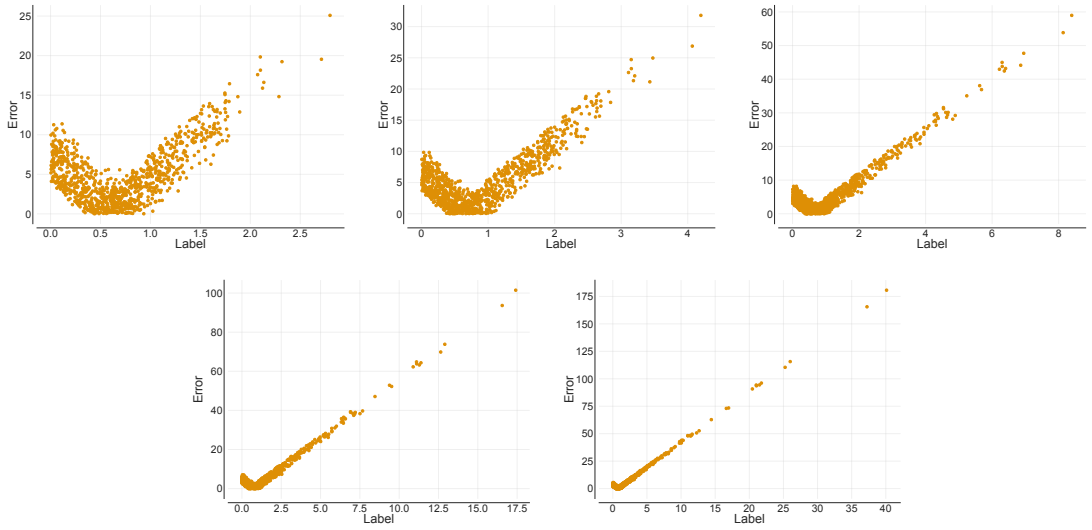

Figure 5.2. Prediction label vs ND error for the different sine datasets. Top Row: Pareto_0.1_3, Pareto_0.1_2, Pareto_0.1_1; Bottom Row: Pareto_0.3_1, Pareto_0.5_1. We can see that the error has some correspondence to the label but with variation. We see higher variance in datasets with shorter tail.

5.1.3 Different-Gaussian Datasets

These datasets are designed to generate histories more complex than the sinuosoidal ones. More complex histories introduces further variation in error for similar labels reducing the direct relation between the label and error. This is illustrated in the prediction label vs ND error plots shown in Figure 5.3.

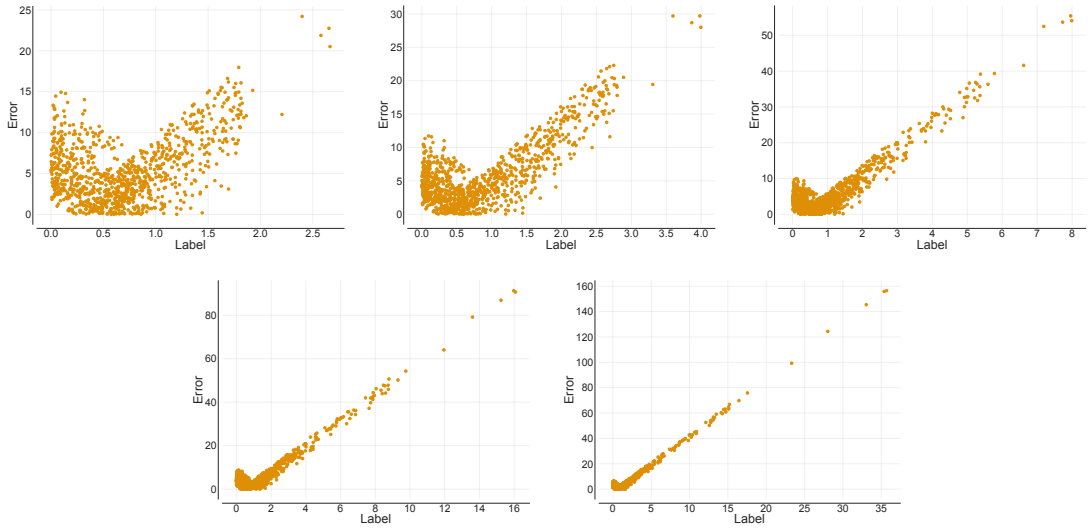

Figure 5.3. Prediction label vs ND error for the different gaussian datasets. Top Row: Pareto_0.1_3, Pareto_0.1_2, Pareto_0.1_1; Bottom Row: Pareto_0.3_1, Pareto_0.5_1. The correspondence of error to the label while still there is much lower. We see much higher variance in datasets with shorter tail.

5.1.4 Correlating Long tails in Data and Error

We are looking to quantify the impact of long tail in data on the long tail in error. This quantification can be achieved by computing correlation between the two. The correlation between the pdf of the prediction label and the corresponding error would represent this relation. Rarer samples or ones with lower pdf should have higher errors and vice versa. This relation however is not necessarily linear. This leads to the choice of computing the Spearman correlation coefficient between the pdf of the labels and the ND error. Since we expect error to increase

for lower pdf values the correlation will be negative and more negative values represent higher correlation. The results of this analysis are present in Table 5.1

Table 5.1. Spearman correlation coefficient values between label pdf and ND error for 1-step forecast datasets. The correlation is higher for longer tail datasets and reduces as we make the history more complex.

DISTRIBUTION NAME	SAME-HISTORY	DIFFERENT-SINE	DIFFERENT-GAUSSIAN
PARETO_0.1_3	-0.192	-0.232	-0.222
PARETO_0.1_2	-0.371	-0.379	-0.475
PARETO_0.1_1	-0.419	-0.427	-0.514
PARETO_0.3_1	-0.505	-0.493	-0.454
PARETO_0.5_1	-0.798	-0.774	-0.628

We can see that the correlation between pdf and error increases as the length of the tail increases. We can also see that different-gaussian has significantly lower correlation in the longer tail datasets than different-sine or same-history. This shows that with complex histories which are typical for real world datasets we can expect long tail in data to explain some part of the long tail in error. However, most correlations are lower than 0.5 in magnitude indicating that there are other factors contributing to the long tail in error as well.

This is the reason why we choose to mitigate the long tail in error directly rather than attempt improvements by adjusting for the long tail in data. The long tail in data is not only hard to identify in complex settings like time series forecasting but as illustrated by our experiments not sufficient to understand the problem either. Approaches focusing on long tail error are agnostic to the cause of the error and work to reduce the long tail in error irrespective. In contrast, methods focusing on long tail in data limit the improvements to only the long tail in error caused by the long tail in data but not the other factors.

Table 5.2. Performance on **Electricity Dataset** (ND/NRMSE/CRPS). All our methods improve on the average as well as tail metrics. Baseline methods are worse on average and inconsistent on the tail. All methods use DeepAR as the base model. Results indicated as Top 3 and **Best**. All results have been averaged across 3 runs with different seeds, standard deviation available in Appendix I

METHOD	METRIC	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
BASE MODEL	ND	0.0600	0.0793	0.2251	0.4356	4.2777
	NRMSE	0.3069	**0.0991**	0.2533	0.5430	5.5994
	CRPS	142	463	1138	1996	30705
CONTRASTIVE LOSS	ND.	0.0696	0.0954	0.2419	0.4646	4.5286
	NRMSE	0.3345	0.1138	0.2778	0.5504	5.6761
	CRPS	167	521	1266	2363	31835
FOCAL LOSS	ND	0.0639	0.0859	0.2505	0.4456	4.3217
	NRMSE	0.3110	0.1062	0.2922	0.5342	5.4843
	CRPS	150	474	1195	2103	30224
SHRINKAGE LOSS	ND	0.0673	0.0888	0.2328	0.4568	4.5911
	NRMSE	0.3247	0.1103	0.2871	0.5213	5.6334
	CRPS	156	480	1199	2240	28398
LDS	ND	0.0632	0.0920	0.2287	0.4620	3.8626
	NRMSE	0.2980	0.1152	0.2790	0.5322	5.0126
	CRPS	151	496	1185	2110	29959
KURTOSIS LOSS (OURS)	ND	**0.0578**	0.0827	0.2132	0.4044	3.6565
	NRMSE	0.2801	0.1023	0.2564	0.4958	4.7673
	CRPS	**140**	455	1105	**1952**	26946
PLM (OURS)	ND	0.0580	**0.0791**	**0.2018**	0.3990	3.7827
	NRMSE	0.2897	0.1011	**0.2396**	**0.4844**	5.0230
	CRPS	141	**449**	1111	2044	28992
PLW (OURS)	ND	0.0581	0.0793	0.2191	**0.3917**	**3.5673**
	NRMSE	**0.2789**	0.1013	0.2569	**0.4973**	**4.7328**
	CRPS	**140**	454	**1099**	1953	**26273**

5.2 Real World Experiments

5.2.1 Cross-task consistency

As shown in Tables 5.2, 5.3, 5.4 and 5.5, our proposed approaches, Kurtosis Loss and PLM, are the only methods improving on tail metrics across all tasks. Our methods typically

Table 5.3. Performance on the **Traffic Dataset** (ND/NRMSE/CRPS). PLM (Ours) delivers best overall results, improving on average and tail metrics. Among baseline methods, contrastive loss is most consistent. Regularization methods in general fare better than re-weighting methods due to a very long tail. All methods use DeepAR as the base model. Results indicated as Top 3 and **Best**

METHOD	METRIC	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
BASE MODEL	ND	0.1741	**0.6866**	25.5840	32.1330	84.1582
	NRMSE	**0.4465**	**1.2283**	6.0283	7.5988	18.8103
	CRPS	0.0068	0.0211	0.0412	0.0691	0.8524
CONTRASTIVE LOSS	ND	0.2052	0.7463	**24.3737**	30.5117	81.1716
	NRMSE	0.4667	1.2956	5.7747	7.2342	18.3360
	CRPS	0.0079	0.0235	0.0450	0.0802	0.8517
FOCAL LOSS	ND	0.4903	1.1553	26.7537	**30.1506**	**52.8272**
	NRMSE	0.7302	1.6485	6.5880	7.3660	13.7985
	CRPS	0.0183	0.0463	0.0639	0.0933	0.8471
SHRINKAGE LOSS	ND	0.2431	0.8380	25.3381	32.9147	85.2713
	NRMSE	0.5114	1.3099	6.0418	7.8882	19.0771
	CRPS	0.0093	0.0316	0.0511	0.0732	0.8573
LDS	ND	0.4763	1.4781	28.9162	38.4263	126.5733
	NRMSE	0.7829	1.8702	6.8826	9.2061	27.3684
	CRPS	0.0175	0.0564	0.0802	0.1074	0.8530
KURTOSIS LOSS (OURS)	ND	0.2022	0.7653	25.3752	31.4677	62.9173
	NRMSE	0.4892	1.4072	6.0263	7.3369	**13.7783**
	CRPS	0.0081	0.0243	**0.0409**	**0.0682**	0.8491
PLM (OURS)	ND	**0.1594**	0.7115	24.5911	30.331	90.3169
	NRMSE	0.4600	1.3881	**5.6779**	**7.0033**	20.5736
	CRPS	**0.0065**	**0.0185**	0.0429	0.0822	**0.8463**
PLW (OURS)	ND	0.3751	1.0495	25.4471	31.6621	65.759
	NRMSE	0.6238	1.4914	6.0552	7.3491	13.8938
	CRPS	0.0126	0.0361	0.0501	0.0716	0.8571

deliver 10-15% improvement on tail metrics and sometimes as high as 40% (See Appendix H). These are significant improvements with no sacrifice on average performance for any task. In fact, in some tasks our methods have better average performance as well.

The generality of our methods is shown by their success on all studied tasks. Our tasks have different base models (DeepAR, Trajectron++EWTA), data representations (1D: Time

Table 5.4. Macro-averaged performance on the **ETH-UCY Dataset** (ADE/FDE). Our approaches improve tail performance better than existing baselines. The improvements are most significant for far-future prediction (FDE). PLM improves well across prediction horizon (ADE). All methods utilize Trajectron++EWTA as the base model. Results indicated as Top 3 and **Best**.

METHOD	MEAN↓	VaR$_{95}$ ↓	VaR$_{98}$ ↓	VaR$_{99}$ ↓	MAX↓
BASE MODEL	**0.16**/0.33	0.43/1.05	0.60/1.53	0.76/1.89	1.63/3.95
CONTRASTIVE	0.17/0.34	0.43/1.03	0.62/1.56	0.79/1.89	1.67/4.02
FOCAL LOSS	**0.16**/0.32	0.40/0.89	0.54/1.28	0.66/1.57	1.50/3.50
SHRINKAGE LOSS	**0.16**/0.33	0.43/1.05	0.58/1.50	0.74/1.84	1.66/3.95
LDS	0.17/0.35	0.44/1.04	0.57/1.45	0.78/1.86	1.69/3.85
KURTOSIS LOSS (OURS)	0.17/0.34	0.46/0.98	0.59/1.25	0.67/1.47	**1.22/2.77**
PLM (OURS)	**0.16/0.30**	**0.38/0.81**	**0.52**/1.20	**0.63**/1.49	1.30/3.20
PLW (OURS)	0.21/0.36	0.46/0.84	0.55/**1.08**	**0.63/1.32**	1.25/2.93

Table 5.5. Average performance on the **nuScenes Dataset** (ADE/FDE). Our methods improve tail performance for far-future prediction (FDE) better than existing baselines. All methods utilize Trajectron++EWTA as the base model. Results indicated as Top 3 and **Best**.

METHOD	MEAN↓	VaR$_{95}$ ↓	VaR$_{98}$ ↓	VaR$_{99}$ ↓	MAX↓
BASE MODEL	**0.19**/0.34	0.65/1.49	1.00/2.49	1.32/3.34	7.07/11.42
CONTRASTIVE	**0.19**/0.35	0.65/1.51	1.01/2.58	1.36/3.46	6.82/10.48
FOCAL LOSS	**0.19**/0.33	**0.56**/1.09	0.85/1.95	1.11/2.65	6.55/11.71
SHRINKAGE LOSS	**0.19/0.32**	0.62/1.32	0.96/2.31	1.25/3.17	6.39/10.26
LDS	**0.19/0.32**	0.62/1.26	0.94/2.23	1.20/2.99	**5.20**/10.53
KURTOSIS LOSS (OURS)	0.20/0.38	0.65/1.35	0.85/1.82	1.03/2.27	5.39/**7.52**
PLM (OURS)	**0.19**/0.33	0.62/1.32	0.95/2.31	1.25/3.18	6.10/10.96
PLW (OURS)	0.24/0.37	0.60/**1.00**	**0.82/1.49**	**1.01/2.01**	7.51/9.91

series, 2D: Trajectory), base losses (GaussianNLL for Time series, EWTA for Trajectory), and forecasting horizons. Our methods provide consistent improvement on tail metrics for all tasks. In comparison, Focal Loss performs well on trajectory datasets but fails on time series datasets. Contrastive Loss only performs well on Traffic dataset. LDS and Shrinkage Loss do not compare to the best results for any dataset and perform worse than the base model on the time series datasets.

We illustrate some difficult examples, examples with large errors common across methods,

for all real world datasets in Figure 5.4 to demonstrate the improvement in the quality of forecast for our methods.

5.2.2 Re-weighting vs Regularization

As mentioned in Section 3.2, we can categorize loss modifying methods into two classes: re-weighting (Focal Loss, Shrinkage Loss, LDS and PLW) and regularization (Contrastive Loss, PLM and Kurtosis Loss). Re-weighting multiplies the loss for tail samples with higher weights. Regularization adds higher regularization values for samples with higher loss.

We notice that re-weighting methods perform worse as the long-tail in error worsens. In scenarios with longer tails, the weights of tail samples can be very high. Overemphasizing tail examples might hamper the learning for other samples. Notice the significantly worse average performance of Focal loss for the traffic dataset in Table 5.3. Shrinkage Loss limits this issue by bounding the weights but fails to show tail improvements in longer tail scenarios (electricity and traffic datasets). Our proposed PLW is the best reweighting method on most datasets, likely due to bounded weights.

In contrast, regularization methods are consistent across all tasks on both tail and average metrics. The additive nature of regularization limits the impact tail samples have on the learning. This enables these methods to handle different severities of long-tail without degrading the average performance.

5.2.3 Choosing between PLM and Kurtosis Loss

Kurtosis Loss performs better on extreme tail metrics, VaR_{99} and Max. Higher kurtosis puts more emphasis on extreme samples in the tail. It is also important to note that the magnitude of kurtosis varies significantly for different distributions, making the choice of hyperparameter (See (3.5)) critical. Further analysis available in Appendix D.

PLM is the most consistent method across all tasks. As noted by [McNeil, 1997] GPD is well suited to model long tail error distributions. PLM rewards examples moving away from the

tail towards the mean with significantly lower margin values. PLM margin values saturate beyond a point in the tail providing similar penalties for long-tail samples. Comparatively, Kurtosis Loss is sensitive to extreme samples in the tail. This shows in performance with Kurtosis Loss performing better on VaR_{99} and Max, and PLM performing better on VaR_{95} and VaR_{98}. The choice between the methods depends on the objective. If the preference is to mitigate extreme samples, then Kurtosis Loss is better. Otherwise, if the preference is to improve on the tail overall, then PLM is better.

5.2.4 Tail error and long-term forecasting

Based on the trajectory prediction results in Tables 5.4 and 5.5 we observe that the error reduction for tail samples is more visible in FDE than in ADE. This indicates that the magnitude of the observed error increases with the forecasting horizon. The error compounds through prediction steps making far-future predictions inherently more difficult. Larger improvements in FDE indicate that both Kurtosis and Pareto Loss ensure that high tail errors (stemming from large, far-future prediction errors measured by FDE) are decreased.

Accurate long-term forecasting is a central goal of deep probabilistic forecasting. As we can see in Figure 5.5, the tail of error distribution is more pronounced with longer horizons. Thus, methods addressing the tail performance are necessary in order to ensure the practical applicability and reliability of future long-term prediction models.

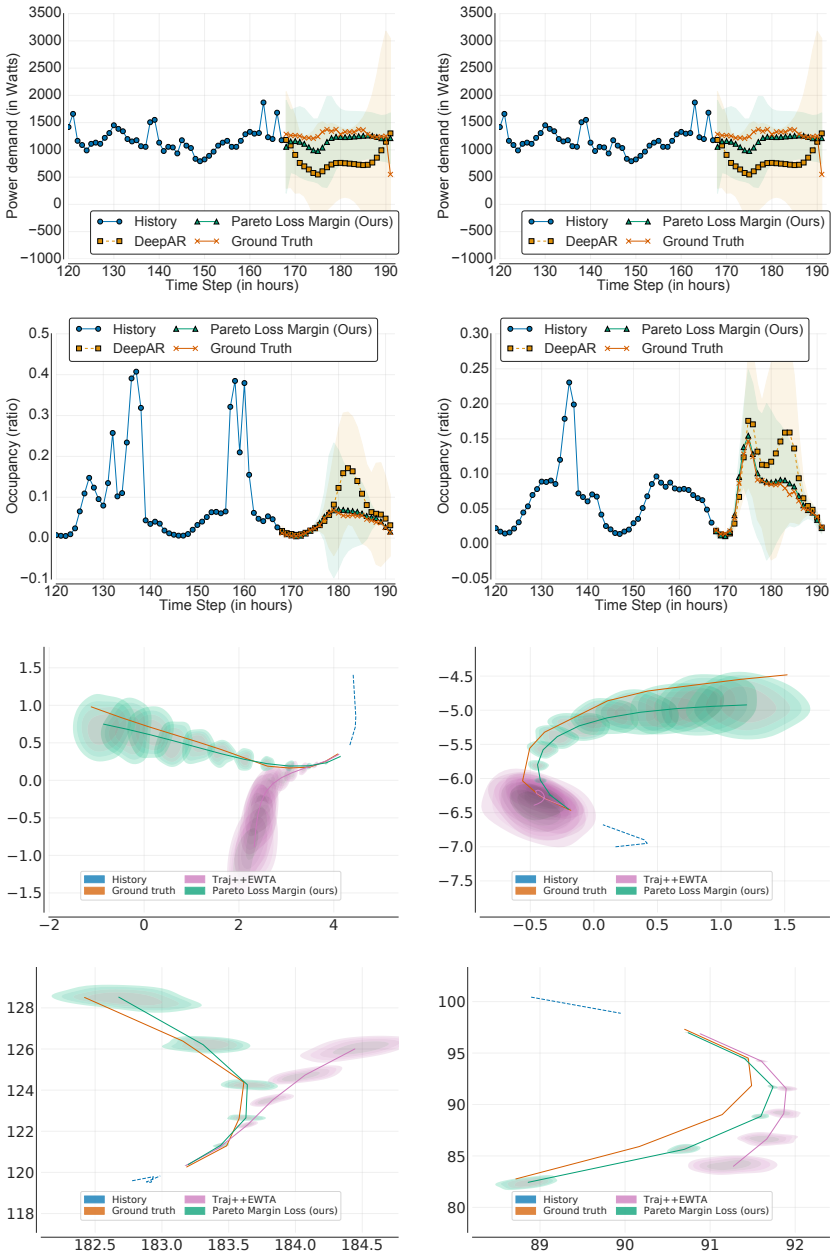

Figure 5.4. Visualization of overlapping tail samples for Electricity (top row left half), Traffic (top row right half), ETH-UCY (bottom row left half) and nuScenes (bottom row right half) datasets. The shaded region represents the confidence interval of the prediction. The difficulty here is a departure from historical behavior. This manifests as sudden increases or decreases in the 1D time series datasets and as high velocity trajectories with sharp turns for the trajectory datasets. These samples represent critical events in real world scenarios where the performance of the model is of utmost importance. Our methods perform significantly better on such samples.

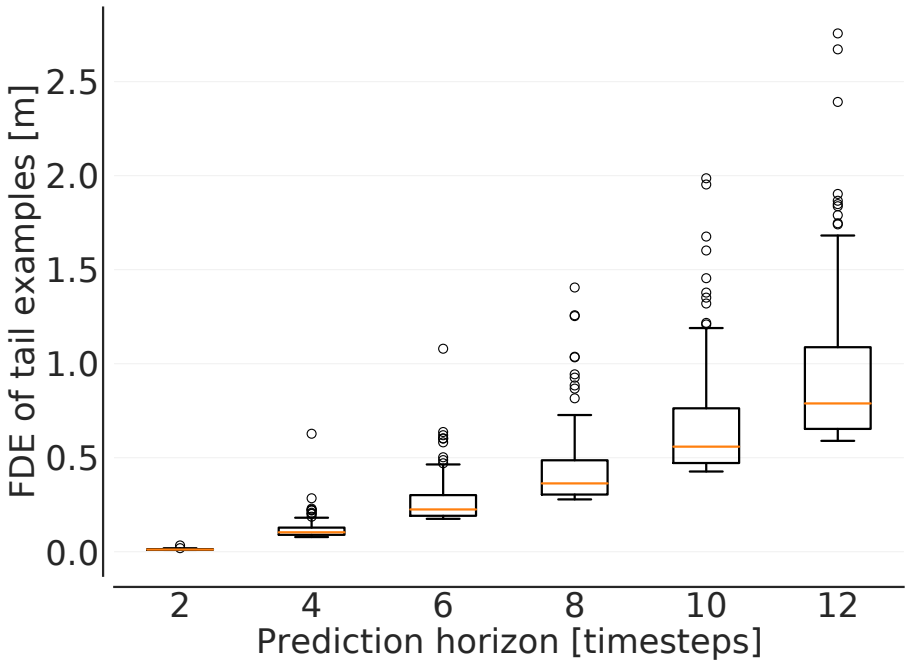

Figure 5.5. Distribution of the top 5% error values (FDE) for different horizons for the ETH-UCY (Zara1) dataset. Predictions obtained using Trajectron++EWTA. The trend shows that the long tail in error gets worse as the forecasting horizon increases due to compounding.

Chapter 6

Conclusion

We identify and address the problem of long-tail in error distribution for deep probabilistic forecasting. We propose Pareto Loss (Margin and Weighted) and Kurtosis Loss, two novel moment-based loss augmentation approaches, increasing emphasis on tail samples adaptively. We demonstrate their practical effects on two spatiotemporal trajectory datasets and two time series datasets using different base models. Our methods achieve significant improvements on tail metrics over existing baselines without degrading average performance. Both proposed losses can be easily integrated with existing approaches in deep probabilistic forecasting to improve their performance on tail metrics.

Future directions include more principled ways to tune hyperparameters, extensions to deterministic time series forecasting models, and theoretical analysis for the source of the long-tail error. Based on our observations, we suggest evaluating tail metrics apart from average performance in machine learning tasks to identify potential long tail issues across different tasks and domains.

Appendix A

Dataset description

The ETH-UCY dataset consists of five subdatasets, each with Bird's-Eye-Views: ETH, Hotel, Univ, Zara1, and Zara2. We present macro-averaged 5-fold cross-validation results in our experiment section, as is common in the literature [Makansi et al., 2021, Salzmann et al., 2020]. The nuScenes dataset includes 1000 scenes of 20 second length for vehicle trajectories recorded in Boston and Singapore.

The electricity dataset contains electricity consumption data for 370 homes over the period of Jan 1st, 2011 to Dec 31st, 2014 at a sampling interval of 15 minutes. We use the data from Jan 1st, 2011 to Aug 31st, 2011 for training and data from Sep 1st, 2011 to Sep 7th, 2011 for testing. The traffic dataset consists of occupancy values recorded by 963 sensors at a sampling interval of 10 minutes ranging from Jan 1st, 2008 to Mar 30th, 2009. We use data from Jan 1st, 2008 to Jun 15th, 2008 for training and data from Jun 16th, 2008 to Jul 15th, 2008 for testing. Both time series datasets are downsampled to 1 hour for generating examples.

The synthetic datasets are generated as 100 different time series consisting of 960 time steps. Each time series in the Sine dataset is generated using a random offset θ and a random frequency ν both selected from a uniform distribution $U[0,1]$. Then the time series is $sin(2\pi\nu t + \theta)$ where t is the index of the time step. Gaussian and Pareto datasets are generated as order 1 lag autoregressive time series with randomly sampled Gaussian and Pareto noise respectively. Gaussian noise is sampled from a Gaussian distribution with mean 1 and standard

deviation 1. Pareto noise is randomly sampled from a Pareto distribution with shape 10 and scaling 1.

Appendix B

Method adaptation

Time Series forecasting

DeepAR uses Gaussian Negative Log Likelihood as the loss which is unbounded. Due to this many baseline methods need to be adapted in order to be usable. For the same reason, we also need an auxiliary loss ($\hat{l}$). We use MAE loss to fit the GPD, calculate kurtosis, and to calculate the weight terms for Focal and Shrinkage loss. For LDS we treat all labels across time steps as a part of a single distribution. Additionally, to avoid extremely high weights ($\mathcal{O}(10^8)$) in LDS due to the nature of long tail we ensure a minimum probability of 0.001 for all labels.

Trajectory forecasting

We adapt Focal Loss and Shrinkage Loss to use EWTA loss [Makansi et al., 2019] in order to be compatible with Trajectron++EWTA base model. LDS was originally proposed for a regression task and we adapt it to the trajectory prediction task in the same way as for the time series task. We use MAE to fit the GPD, due to the Evolving property of EWTA loss.

Appendix C

Implementation details

Time Series forecasting

We use the DeepAR implementation from https://github.com/zhykoties/TimeSeries as the base code to run all time series experiments. The original code is an AWS API and not publicly available. The implementation of contrastive loss is taken directly from the source code of [Makansi et al., 2021].

Trajectory forecasting

For the base model of Trajectron++EWTA [Makansi et al., 2021] we have used the original implementation provided by the original authors. The implementation of contrastive loss is taken directly from the source code of [Makansi et al., 2021].

The experiments have been conducted on a machine with 7 RTX 2080 Ti GPUs.

Appendix D

Hyperparameter Tuning

We observe during our experiments that the performance of Kurtosis Loss is highly dependent on the hyperparameter λ (See (3.5)). Results for different values of λ on the electricity dataset for Kurtosis Loss are shown in TableD.1. We also show the variation of ND and NRMSE with the hyperparameter value in Figure D.1. We can see that there is an optimal value of the hyperparameter and the approach performs worse with higher and lower values.

For both ETH-UCY and nuScenes datasets we have used $\lambda = 0.1$ for Kurtosis Loss, and $\lambda = 1$ for PLM and PLW. For both electricity and traffic datasets, we use $\lambda = 1$ for PLM, $\lambda = 0.5$ for PLW and $\lambda = 0.01$ for Kurtosis Loss.

Table D.1. Electricity Dataset evaluation for base model (ND/NRMSE) and different Kurtosis Loss hyperparameters. The value of λ is denoted in [] with the method name. The base model is DeepAR. Results indicated as **Best** and Better than base model

METHOD	METRIC	MEAN↓	VAR_{95} ↓	VAR_{98} ↓	VAR_{99} ↓	MAX↓
BASE MODEL	ND	0.0584	0.0796	0.2312	0.4429	4.1520
	NRMSE	0.2953	**0.0972**	0.2595	0.5263	5.4950
KURTOSIS LOSS [0.001]	ND	0.0581	0.0815	**0.2087**	**0.3936**	4.2381
	NRMSE	0.3046	0.1014	**0.2325**	**0.4756**	5.7144
KURTOSIS LOSS [0.005]	ND	0.0574	**0.0767**	0.2147	0.4138	3.6767
	NRMSE	0.2843	0.0999	0.2617	0.4792	5.0062
KURTOSIS LOSS [0.01]	ND	**0.0567**	0.0842	0.2151	0.4120	**3.2738**
	NRMSE	**0.2631**	0.1046	0.2732	0.4779	**4.2613**
KURTOSIS LOSS [0.1]	ND	0.0677	0.0954	0.2269	0.4579	3.8772
	NRMSE	0.3073	0.1184	0.2768	0.5419	5.1345

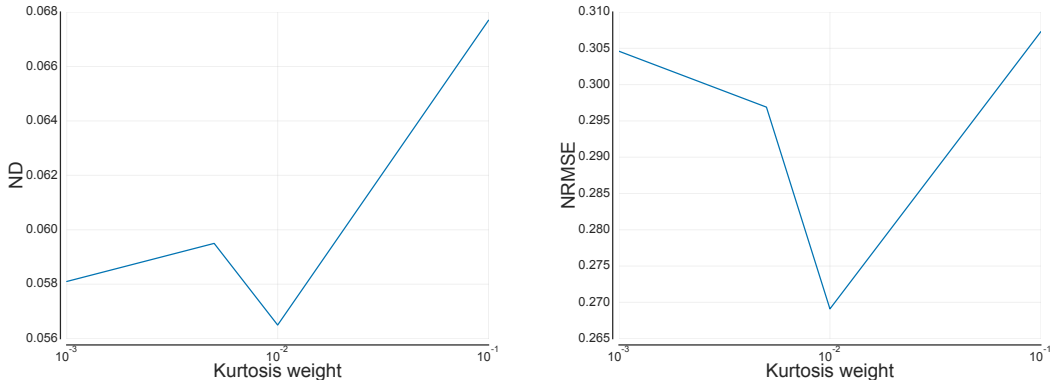

Figure D.1. Left: Variation of ND by hyperparameter for Kurtosis Loss. Right: Variation of NRMSE by hyperparameter for Kurtosis Loss.

Appendix E

Pareto and Kurtosis

Figure E.1 illustrates different GPDs for different shape parameter values. Higher shape value models more severe tail behavior.

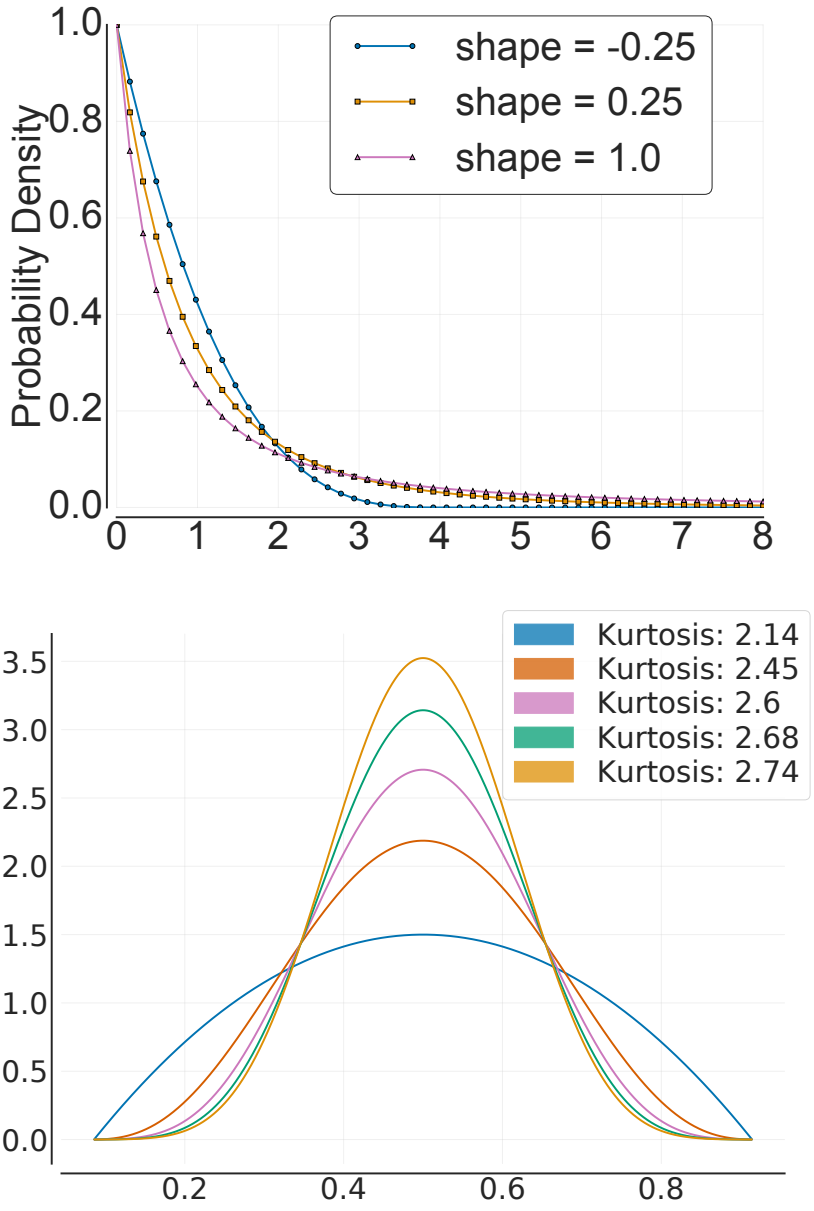

Figure E.1. Left: Generalized Pareto distributions with different shape parameters ($\eta = 1$). Right: Illustrating the variation of kurtosis on distributions with the same mean.

Appendix F

Auxiliary loss

In this section, we present mathematical intuition behind the usage of auxiliary loss in our methods. We will examine a setting where the base loss for a probabilistic model is GaussianNLL loss and the evaluation metric is MSE. For simplicity, we will assume 1-step prediction on 1D data however the analysis can be easily extended to multi step prediction and multi dimensional data.

Consider 2 training samples,

Past observations : $\mathbf{x}^{(1)} = (x_{t-k}^{(1)}, \ldots, x_t^{(1)})$; $\mathbf{x}^{(2)} = (x_{t-k}^{(2)}, \ldots, x_t^{(2)})$

1-step prediction ground truth : $\mathbf{y}^{(1)} = (y_{t+1}^{(1)})$; $\mathbf{y}^{(2)} = (y_{t+1}^{(2)})$

Model prediction : $\mu_{t+1}^{(1)}, \sigma_{t+1}^{(1)}; \mu_{t+1}^{(2)}, \sigma_{t+1}^{(2)}$

We will drop t+1 from the notation for simplicity and clarity as there is only one step prediction. Since, the maximum likelihood prediction for a gaussian is the mean, the MSE is calculated using the predicted mean.

$$\text{MSE} : (y^{(i)} - \mu^{(i)})^2 \tag{F.1}$$

The GaussianNLL loss is calculated as the negative log likelihood of the ground truth as per the predicted distribution. Simplifying the expression gives us,

$$\text{GaussianNLL loss} : \ln(\sigma^{(i)}\sqrt{2\pi}) + \frac{1}{2}\left(\frac{y^{(i)} - \mu^{(i)}}{\sigma^{(i)}}\right)^2 \tag{F.2}$$

We want to determine the conditions under which the GaussianNLL loss will be higher for sample 1 as compared to sample 2 while the MSE for sample 2 will be higher than sample 1 or vice versa. We will call this a loss-metric inversion. This condition can be written as:

$$(GaussianNLL^{(1)} - GaussianNLL^{(2)})(MSE^{(1)} - MSE^{(2)}) < 0 \tag{F.3}$$

Consider the scenario where, $MSE^{(1)} > MSE^{(2)}$. This can be expressed as,

$$(y^{(1)} - \mu^{(1)})^2 = k(y^{(2)} - \mu^{(2)})^2 \text{ where } k > 1 \qquad \text{(From (F.1))} \tag{F.4}$$

The corresponding condition to satisfy is,

$$\begin{aligned}
&(GaussianNLL^{(1)} - GaussianNLL^{(2)}) < 0 && \text{(From (F.3))} \\
&\implies \ln(\frac{\sigma^{(1)}}{\sigma^{(2)}}) + \frac{1}{2}\left(\left(\frac{y^{(1)} - \mu^{(1)}}{\sigma^{(1)}}\right)^2 - \left(\frac{y^{(2)} - \mu^{(1)}}{\sigma^{(2)}}\right)^2\right) < 0 && \text{(From (F.2))} \\
&\implies \frac{1}{2}(y^{(2)} - \mu^{(2)})^2\left(\frac{k}{\sigma^{(1)^2}} - \frac{1}{\sigma^{(2)^2}}\right) < \ln(\frac{\sigma^{(2)}}{\sigma^{(1)}}) && \text{(From (F.4))}
\end{aligned}$$

Consider, $\sigma^{(1)} = c\sigma^{(2)}$, where $c > 0$

$$\frac{1}{2}\left(\frac{y^{(2)} - \mu^{(2)}}{\sigma^{(2)}}\right)^2\left(\frac{k}{c^2} - 1\right) < \ln(\frac{1}{c})$$

For simplicity let's represent $\frac{1}{2}\left(\frac{y^{(2)} - \mu^{(2)}}{\sigma^{(2)}}\right)^2$ as a single variable m.

$$m\left(\frac{k}{c^2} - 1\right) + \ln(c) < 0$$

For a fixed k the minima for the LHS is achieved for $c = \sqrt{2km}$. The value of the LHS at minima is,

$$\left(\frac{1}{2}-m\right)+\frac{1}{2}\ln(km)=\frac{1}{2}\ln\left(\frac{km}{e^{2m-1}}\right)$$

Since the numerator in the log form is linear in m and the denominator is exponential in m the minima can be less than zero for suitable values of m.

This shows that there can be pairs of samples with loss-metric inversion. This means that regularization and reweighting values can be completely different from intended unless an auxiliary loss is used, which preserves the order w.r.t. the evaluation metric. This lack of correlation is illustrated in Fig F.1 for the DeepAR model on the electricity dataset.

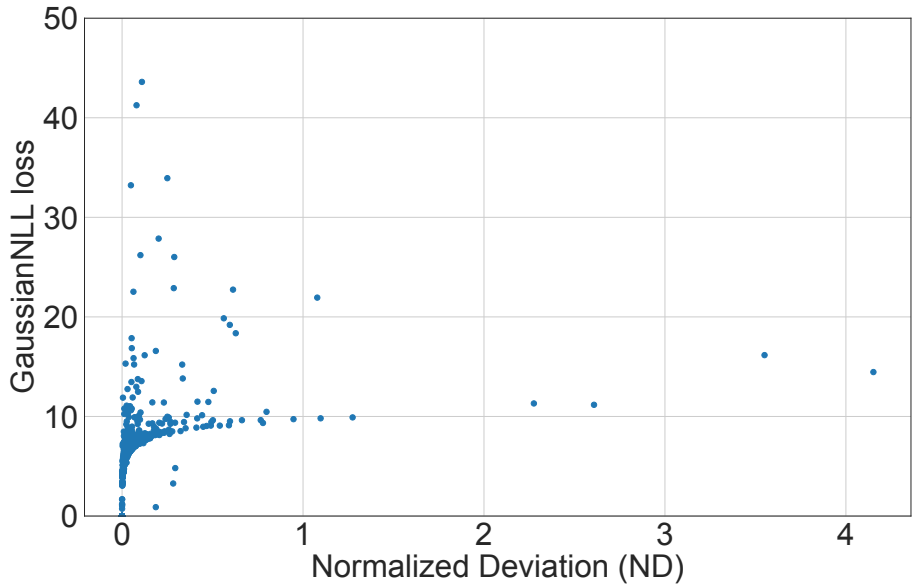

Figure F.1. Comparing GaussianNLL loss to Normalized Deviation metric for DeepAR on the electricity dataset. We can see that there are a large number of samples which have high GaussianNLL but low ND and vice versa. This illustrates the need of an auxiliary loss for correct emphasis on samples.

Appendix G

1-Step Forecast Dataset Generation

For each dataset we generated 4000 training samples and 1000 test samples. Each sample consisted of 7 history points generated using either a sinusoidal process or

G.1 Sinusoidal History Generation

Sinusoidal time series is quite simple to generate. We randomly sample parameters from uniform distribution and then use a mixture of sin and cos with those parameters to generate the data at different timesteps. The parameters and their ranges are:

- $4 \leq$ period (pd) ≤ 20
- $0 \leq$ phase (ph) $\leq \pi$
- $0 \leq$ weight (w) ≤ 1

The generation equation is,

$$x_t = w(\sin(\frac{2\pi t}{pd} + ph)) + (1-w)(\cos(\frac{2\pi t}{pd} + ph))$$

For the same-history datasets we generate a single history with a randomly picked set of parameters while for the different-sine datasets we generate a different set of parameters for each series to be generated.

G.2 Gaussian History Generation

Gaussian time series data is generated as follows:

$$x_{t+1} = 0.5 * x_t + gs$$

where

$$gs \sim \mathcal{N}(1,1)$$

We set the mean to be 1 so that most sampled values are positive. Additionally, to generate the first point in the time series we use the same equation where as we do not have x_{-1} we instead use a randomly picked value from a uniform distribution on $[0,1]$

G.3 Pareto Label Generation

The labels for all datasets are sampled from a pareto distribution. This is relatively straightforward. For a given shape ξ, we use the pareto distribution with pdf,

$$f(x,\xi) = (1+\xi x)^{-\frac{\xi+1}{\xi}}$$

For different ratios, to reduce the length of the tail we divide all samples greater than a threshold (much higher than the median) by the ratio. This reduces the length of the tail without altering the median much or significantly changing the shape of the distribution.

Appendix H

Percentage Improvements

We present percentage improvements compared to the base model for the different datasets.

Table H.1. Percentage improvements over the base method (DeepAR) on **Electricity Dataset** (ND/NRMSE). Results indicated as error reduction (green) and increase (red) in %.

METHOD	METRIC	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
CONTRASTIVE LOSS	ND	15.94	20.26	7.43	6.67	5.86
	NRMSE	8.99	14.79	9.66	1.24	1.37
FOCAL LOSS	ND	6.44	8.32	11.27	2.30	1.03
	NRMSE	1.34	7.16	15.34	1.72	2.06
SHRINKAGE LOSS	ND	12.17	11.94	3.42	4.86	7.33
	NRMSE	5.80	11.26	13.33	4.10	0.61
LDS	ND	5.28	16.06	1.57	6.06	9.71
	NRMSE	2.90	16.21	10.13	2.10	10.48
KURTOSIS LOSS (OURS)	ND	3.72	4.33	5.29	7.16	14.52
	NRMSE	8.72	3.16	1.21	8.80	14.86
PLM (OURS)	ND	3.28	0.21	10.36	8.40	11.57
	NRMSE	5.58	1.98	5.41	10.89	10.29
PLW (OURS)	ND	3.22	0.00	2.68	10.09	16.61
	NRMSE	9.10	2.19	1.42	8.52	15.48

Table H.2. Percentage improvements over the base method (DeepAR) on **Traffic Dataset** (ND/NRMSE). Results indicated as error reduction (green) and increase (red) in %.

METHOD	METRIC	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
CONTRASTIVE LOSS	ND	17.86	8.70	4.73	5.05	3.55
	NRMSE	4.52	5.48	4.21	4.80	2.52
FOCAL LOSS	ND	181.62	68.26	4.57	6.17	37.23
	NRMSE	63.54	34.21	9.28	3.06	26.64
SHRINKAGE LOSS	ND	39.63	22.05	0.96	2.43	1.32
	NRMSE	14.54	6.64	0.22	3.81	1.42
LDS	ND	173.58	115.28	13.02	19.59	50.40
	NRMSE	75.34	52.26	14.17	21.15	45.50
KURTOSIS LOSS (OURS)	ND	16.14	11.46	0.82	2.07	25.24
	NRMSE	9.56	14.56	0.03	3.45	26.75
PLM (OURS)	ND	8.44	3.63	3.88	5.61	7.32
	NRMSE	3.02	13.01	5.81	7.84	9.37
PLW (OURS)	ND	115.45	52.85	0.54	1.47	21.86
	NRMSE	39.71	21.42	0.45	3.29	26.14

Table H.3. Percentage improvements over the base method (Trajectron++EWTA) on **ETH-UCY Dataset** (ADE/FDE). Results indicated as error reduction (green) and increase (red) in %.

METHOD	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
CONTRASTIVE	6.25/3.03	0.00/1.90	3.33/1.96	3.95/0.00	2.45/1.77
FOCAL LOSS	0.00/3.03	6.98/15.24	10.00/16.34	13.16/16.93	7.98/11.39
SHRINKAGE LOSS	0.00/0.00	0.00/0.00	3.33/1.96	2.63/2.65	1.84/0.00
LDS	6.25/6.06	2.33/0.95	5.00/5.23	2.63/1.59	3.68/2.53
KURTOSIS LOSS (OURS)	6.25/3.03	6.98/6.67	1.67/18.30	11.84/22.22	25.15/29.87
PLM (OURS)	0.00/9.09	11.63/22.86	13.33/21.57	17.11/21.16	20.25/18.99
PLW (OURS)	31.25/9.09	6.98/20.00	8.33/29.41	17.11/30.16	23.31/25.82

Table H.4. Percentage improvements over the base method (Trajectron++EWTA) on **nuScenes Dataset** (ADE/FDE). Results indicated as error reduction (green) and increase (red) in %.

METHOD	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
CONTRASTIVE	0.00/2.94	0.00/1.34	1.00/3.61	3.03/3.59	3.54/8.23
FOCAL LOSS	0.00/2.94	13.85/26.85	15.00/21.69	15.91/20.66	7.36/2.54
SHRINKAGE LOSS	0.00/5.88	4.62/11.41	4.00/7.23	5.30/5.09	9.62/10.16
LDS	0.00/5.88	4.62/15.44	6.00/10.44	9.09/10.48	26.45/7.79
KURTOSIS LOSS (OURS)	5.26/11.76	0.00/9.40	15.00/26.91	21.97/32.04	23.76/34.15
PLM (OURS)	0.00/2.94	4.62/11.41	5.00/7.23	5.30/4.79	13.72/4.03
PLW (OURS)	26.32/8.82	7.69/32.89	18.00/40.16	23.48/39.82	6.22/13.22

Appendix I

Electricity Dataset Standard Deviation

Due to space limitations we were not able to report std dev across the 3 runs for the electricity dataset. We present the same in Table I.1.

Table I.1. Std deviation of results for **Electricity Dataset** (ND/NRMSE/CRPS). All results have been computed across 3 runs with different seeds. Results corresponding to Table 5.2.

METHOD	METRIC	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
BASE MODEL	ND	0.0023	0.0024	0.0060	0.0258	0.1092
	NRMSE	0.0102	0.0033	0.0057	0.0407	0.0919
	CRPS	2	13	27	69	178
CONTRASTIVE LOSS	ND	0.0075	0.0110	0.0276	0.0425	0.5142
	NRMSE	0.0296	0.0108	0.0294	0.0195	0.5382
	CRPS	17	41	120	184	1033
FOCAL LOSS	ND	0.0018	0.0010	0.0164	0.0203	0.1247
	NRMSE	0.0067	0.0009	0.0189	0.0221	0.3053
	CRPS	4	27	50	68	1990
SHRINKAGE LOSS	ND	0.0021	0.0059	0.0134	0.0248	0.3039
	NRMSE	0.0124	0.0048	0.0048	0.0038	0.4650
	CRPS	5	12	15	170	2826
LDS	ND	0.0014	0.0048	0.0054	0.0401	0.7368
	NRMSE	0.0249	0.0074	0.0068	0.0350	0.8518
	CRPS	5	26	29	81	4051
KURTOSIS LOSS (OURS)	ND	0.0010	0.0034	0.0084	0.0145	0.3646
	NRMSE	0.0153	0.0039	0.0179	0.0170	0.4872
	CRPS	3	15	44	54	2845
PLM (OURS)	ND	0.0021	0.0009	0.0119	0.0308	0.3861
	NRMSE	0.0129	0.0010	0.0047	0.0225	0.5220
	CRPS	3	10	48	83	2205
PLW (OURS)	ND	0.0013	0.0026	0.0183	0.0311	0.1256
	NRMSE	0.0047	0.0026	0.0164	0.0154	0.1142
	CRPS	3	8	10	57	1215

Appendix J

Training details

The training procedure employed for the Pareto Losses is as follows:

- Train the base model until convergence
- Fit the Pareto distribution on the loss distribution from the trained model. This is done on the auxiliary loss if one is being used.
- Use the fitted Pareto distribution to implement PLM or PLW and retrain the model.
- The retrained model is the one employing PLM or PLW as per choice.

The training process for Kurtosis loss is straightforward. We use the loss function in Equation (5) directly with one round of training.

Appendix K

Robust Statistics Methods

We ran robust regression methods on the task and found that the results do not show improvements on the long tail of error. The methods examined here are Huber Loss and MSLE.

Table K.1. Results for robust statistics losses on the **Electricity dataset**. Results indicated as **Best**. Huber Loss and MSLE both fail to provide any meaningful improvements on the base model. Moreover, the performance on CRPS is significantly worse illustrating their poor fit for the task.

METHOD	METRIC	MEAN↓	VaR_{95} ↓	VaR_{98} ↓	VaR_{99} ↓	MAX↓
BASE MODEL	ND	0.0600	0.0793	0.2251	0.4356	4.2777
	NRMSE	0.3069	**0.0991**	0.2533	0.5430	5.5994
	CRPS	142	463	1138	**1996**	30705
HUBER LOSS	ND	0.0594	0.0822	0.2378	0.4296	3.7959
	NRMSE	0.2981	0.1041	0.2492	0.5393	5.3614
	CRPS	544	1792	4892	8898	31001
MSLE	ND	0.0608	0.0826	0.2434	0.4336	3.9035
	NRMSE	0.3092	0.1162	0.2993	0.5753	5.2328
	CRPS	601	1998	5683	9935	29485
PLM (OURS)	ND	**0.0580**	**0.0791**	**0.2018**	**0.3990**	**3.7827**
	NRMSE	**0.2897**	0.1011	**0.2396**	**0.4844**	**5.0230**
	CRPS	**141**	**449**	**1111**	2044	**28992**

Appendix L

Synthetic datasets

We present complete results of our experiments on the synthetic datasets in Table L.1. We ran our methods, Kurtosis Loss, and PLM on these datasets as well. Both our methods show significant tail improvements over the base model across all datasets.

Table L.1. Performance on the Synthetic Datasets (ND/NRMSE). Results indicated as Better than DeepAR and **Best** for each dataset.

METHOD	METRIC	MEAN↓	VAR_{95} ↓	VAR_{98} ↓	VAR_{99} ↓	MAX↓
			SINE DATASET			
AUTOREG	ND	1.2255	2.162	2.7088	2.9306	3.1271
	NRMSE	1.5078	2.3134	2.7204	2.9379	3.1271
DEEPAR	ND	0.0513	0.1721	0.316	0.5913	1.5744
	NRMSE	0.1534	0.2009	0.3507	0.6199	1.654
KURTOSIS LOSS	ND	**0.0455**	0.1412	**0.2914**	**0.4470**	**1.5571**
	NRMSE	**0.1330**	**0.1624**	**0.3455**	**0.5387**	**1.5571**
PARETO LOSS	ND	**0.0462**	**0.1326**	0.3014	0.7151	1.582
MARGIN	NRMSE	0.1517	0.1563	0.3551	0.737	1.7522
			GAUSSIAN DATASET			
AUTOREG	ND	0.5730	1.0225	1.3334	1.6226	27.6956
	NRMSE	1.2705	1.1212	1.4045	1.6815	39.7474
DEEPAR	ND	0.4379	0.7050	**0.7908**	0.8651	1.1362
	NRMSE	0.5518	**0.8172**	0.9246	0.9908	1.3009
KURTOSIS LOSS	ND	**0.4378**	0.7040	0.7973	**0.8597**	1.1294
	NRMSE	**0.5518**	0.8191	0.9255	**0.9865**	1.2951
PARETO LOSS	ND	0.4391	**0.7023**	0.7946	0.8674	**1.1069**
MARGIN	NRMSE	0.5534	0.8194	**0.9232**	0.9889	**1.2786**
			PARETO DATASET			
AUTOREG	ND	1.9377	1.1748	1.7039	2.4782	2113.7503
	NRMSE	81.1652	1.4027	1.9856	2.7312	4069.3972
DEEPAR	ND	0.4416	0.8336	1.0317	1.1763	2.015
	NRMSE	0.6349	1.1511	1.4295	1.6688	2.8327
KURTOSIS LOSS	ND	0.4413	0.8345	**1.0295**	**1.1738**	2.0326
	NRMSE	0.6352	1.1541	1.4305	**1.6653**	2.8335
PARETO LOSS	ND	**0.4394**	0.8497	1.0473	1.1955	2.086
MARGIN	NRMSE	0.6397	1.1694	1.4470	1.6735	2.845

Bibliography

[Bollerslev, 1986] Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of econometrics*, 31(3):307–327.

[Branco et al., 2017] Branco, P., Torgo, L., and Ribeiro, R. P. (2017). Smogn: a pre-processing approach for imbalanced regression. In *First international workshop on learning with imbalanced domains: Theory and applications*, pages 36–50. PMLR.

[Caesar et al., 2020] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631.

[Cao et al., 2019] Cao, K., Wei, C., Gaidon, A., Arechiga, N., and Ma, T. (2019). Learning imbalanced datasets with label-distribution-aware margin loss. In *International Conference on Neural Information Processing Systems*.

[Chang et al., 2019] Chang, M.-F., Lambert, J., Sangkloy, P., Singh, J., Bak, S., Hartnett, A., Wang, D., Carr, P., Lucey, S., Ramanan, D., et al. (2019). Argoverse: 3d tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8748–8757.

[Chawla et al., 2002] Chawla, N. V., Bowyer, K. W., Hall, L. O., and Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357.

[Cheng et al., 2015] Cheng, G., Wang, S., and Yang, Y. (2015). Forecast combination under heavy-tailed errors. *Econometrics*, 3(4):797–824.

[Cui et al., 2019] Cui, Y., Jia, M., Lin, T.-Y., Song, Y., and Belongie, S. (2019). Class-balanced loss based on effective number of samples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9268–9277.

[de Bézenac et al., 2020] de Bézenac, E., Rangapuram, S. S., Benidis, K., Bohlke-Schneider, M., Kurle, R., Stella, L., Hasson, H., Gallinari, P., and Januschowski, T. (2020). Normalizing kalman filters for multivariate time series analysis. In *NeurIPS*.

[Dua and Graff, 2017] Dua, D. and Graff, C. (2017). UCI machine learning repository.

[Fan et al., 2017] Fan, Y., Lyu, S., Ying, Y., and Hu, B.-G. (2017). Learning with average top-k loss. In *International Conference on Neural Information Processing Systems*.

[Feldman, 2020] Feldman, V. (2020). Does learning require memorization? a short tale about a long tail. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 954–959.

[Feldman and Zhang, 2020] Feldman, V. and Zhang, C. (2020). What neural networks memorize and why: Discovering the long tail via influence estimation. In *International Conference on Neural Information Processing Systems*.

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

[Gupta et al., 2018] Gupta, A., Johnson, J., Fei-Fei, L., Savarese, S., and Alahi, A. (2018). Social gan: Socially acceptable trajectories with generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2255–2264.

[Kalman, 1960] Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of Basic Engineering*, 82(1):35–45.

[Kozerawski et al., 2020] Kozerawski, J., Fragoso, V., Karianakis, N., Mittal, G., Turk, M., and Chen, M. (2020). Blt: Balancing long-tailed datasets with adversarially-perturbed images. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*.

[Kreil et al., 2020] Kreil, D. P., Kopp, M. K., Jonietz, D., Neun, M., Gruca, A., Herruzo, P., Martin, H., Soleymani, A., and Hochreiter, S. (2020). The surprising efficiency of framing geo-spatial time series forecasting as a video prediction task–insights from the iarai traffic4cast competition at neurips 2019. In *NeurIPS 2019 Competition and Demonstration Track*, pages 232–241. PMLR.

[Kulik and Soulier, 2020] Kulik, R. and Soulier, P. (2020). *Heavy-tailed time series*. Springer.

[Lee et al., 2017] Lee, N., Choi, W., Vernaza, P., Choy, C. B., Torr, P. H., and Chandraker, M. (2017). Desire: Distant future prediction in dynamic scenes with interacting agents. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 336–345.

[Lerner et al., 2007] Lerner, A., Chrysanthou, Y., and Lischinski, D. (2007). Crowds by example. In *Computer graphics forum*, volume 26, pages 655–664. Wiley Online Library.

[Li et al., 2019] Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, 32:5243–5253.

[Li et al., 2020] Li, Y., Wang, T., Kang, B., Tang, S., Wang, C., Li, J., and Feng, J. (2020). Overcoming classifier imbalance for long-tail object detection with balanced group softmax. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10991–11000.

[Lin et al., 2017] Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988.

[Liu et al., 2016] Liu, W., Wen, Y., Yu, Z., and Yang, M. (2016). Large-margin softmax loss for convolutional neural networks. In *ICML*, volume 2, page 7.

[Liu et al., 2019a] Liu, Y., Yu, R., Zheng, S., Zhan, E., and Yue, Y. (2019a). Naomi: Non-autoregressive multiresolution sequence imputation. *Advances in Neural Information Processing Systems*.

[Liu et al., 2019b] Liu, Z., Miao, Z., Zhan, X., Wang, J., Gong, B., and Yu, S. X. (2019b). Large-scale long-tailed recognition in an open world. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2537–2546.

[Lu et al., 2018] Lu, X., Ma, C., Ni, B., Yang, X., Reid, I., and Yang, M.-H. (2018). Deep regression tracking with shrinkage loss. In *Proceedings of the European conference on computer vision (ECCV)*, pages 353–369.

[Luo et al., 2020] Luo, C., Sun, L., Dabiri, D., and Yuille, A. (2020). Probabilistic multi-modal trajectory prediction with lane attention for autonomous vehicles. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2370–2376. IEEE.

[Makansi et al., 2021] Makansi, O., Cicek, Ö., Marrakchi, Y., and Brox, T. (2021). On exposing the challenging long tail in future prediction of traffic actors. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

[Makansi et al., 2019] Makansi, O., Ilg, E., Cicek, O., and Brox, T. (2019). Overcoming limitations of mixture density networks: A sampling and fitting framework for multimodal future prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7144–7153.

[Makridakis et al., 2020] Makridakis, S., Spiliotis, E., and Assimakopoulos, V. (2020). The m5 accuracy competition: Results, findings and conclusions. *Int J Forecast*.

[McNeil, 1997] McNeil, A. J. (1997). Estimating the tails of loss severity distributions using extreme value theory. *ASTIN Bulletin: The Journal of the IAA*, 27(1):117–137.

[Menon et al., 2020] Menon, A. K., Jayasumana, S., Rawat, A. S., Jain, H., Veit, A., and Kumar, S. (2020). Long-tail learning via logit adjustment. In *International Conference on Learning Representations*.

[Pal et al., 2021] Pal, S., Ma, L., Zhang, Y., and Coates, M. (2021). Rnn with particle flow for probabilistic spatio-temporal forecasting. *ICML*.

[Pan et al., 2021] Pan, T.-Y., Zhang, C., Li, Y., Hu, H., Xuan, D., Changpinyo, S., Gong, B., and Chao, W.-L. (2021). On model calibration for long-tailed object detection and instance segmentation. *Advances in Neural Information Processing Systems*, 34.

[Pellegrini et al., 2009] Pellegrini, S., Ess, A., Schindler, K., and Van Gool, L. (2009). You'll never walk alone: Modeling social behavior for multi-target tracking. In *2009 IEEE 12th International Conference on Computer Vision*, pages 261–268. IEEE.

[Prasad et al., 2018] Prasad, A., Suggala, A. S., Balakrishnan, S., and Ravikumar, P. (2018). Robust estimation via robust gradient estimation. *arXiv preprint arXiv:1802.06485*.

[Rangapuram et al., 2018] Rangapuram, S. S., Seeger, M. W., Gasthaus, J., Stella, L., Wang, Y., and Januschowski, T. (2018). Deep state space models for time series forecasting. *Advances in neural information processing systems*, 31:7785–7794.

[Rasul et al., 2021a] Rasul, K., Seward, C., Schuster, I., and Vollgraf, R. (2021a). Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. *ICML*.

[Rasul et al., 2021b] Rasul, K., Sheikh, A.-S., Schuster, I., Bergmann, U., and Vollgraf, R. (2021b). Multivariate probabilistic time series forecasting via conditioned normalizing flows. *ICLR*.

[Ren et al., 2020] Ren, J., Yu, C., Sheng, S., Ma, X., Zhao, H., Yi, S., and Li, H. (2020). Balanced meta-softmax for long-tailed visual recognition. In *International Conference on Neural Information Processing Systems*.

[Ribeiro and Moniz, 2020] Ribeiro, R. P. and Moniz, N. (2020). Imbalanced regression and extreme value prediction. *Machine Learning*, 109(9):1803–1835.

[Salinas et al., 2020] Salinas, D., Flunkert, V., Gasthaus, J., and Januschowski, T. (2020). Deepar: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting*, 36(3):1181–1191.

[Salzmann et al., 2020] Salzmann, T., Ivanovic, B., Chakravarty, P., and Pavone, M. (2020). Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 683–700. Springer.

[Sohn et al., 2015] Sohn, K., Lee, H., and Yan, X. (2015). Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491.

[Steininger et al., 2021] Steininger, M., Kobs, K., Davidson, P., Krause, A., and Hotho, A. (2021). Density-based weighting for imbalanced regression. *Machine Learning*, 110(8):2187–2211.

[Tang et al., 2020] Tang, K., Huang, J., and Zhang, H. (2020). Long-tailed classification by keeping the good and removing the bad momentum causal effect. In *International Conference on Neural Information Processing Systems*.

[Tang and Salakhutdinov, 2019] Tang, Y. C. and Salakhutdinov, R. (2019). Multiple futures prediction. *NeurIPS*.

[Tiong et al., 2021] Tiong, A. M. H., Li, J., Lin, G., Li, B., Xiong, C., and Hoi, S. C. (2021). Improving tail-class representation with centroid contrastive learning. *arXiv preprint arXiv:2110.10048*.

[Torgo et al., 2013] Torgo, L., Ribeiro, R. P., Pfahringer, B., and Branco, P. (2013). Smote for regression. In *Portuguese conference on artificial intelligence*, pages 378–389. Springer.

[Walters et al., 2021] Walters, R., Li, J., and Yu, R. (2021). Trajectory prediction using equivariant continuous convolution. *International Conference on Learning Representations*.

[Wang et al., 2021] Wang, X., Lian, L., Miao, Z., Liu, Z., and Yu, S. X. (2021). Long-tailed recognition by routing diverse distribution-aware experts. In *International Conference on Learning Representations (ICLR)*.

[Wang et al., 2019] Wang, Y., Smola, A., Maddix, D., Gasthaus, J., Foster, D., and Januschowski, T. (2019). Deep factors for forecasting. In *International conference on machine learning*, pages 6607–6617. PMLR.

[Xu et al., 2021] Xu, Z., Chai, Z., and Yuan, C. (2021). Towards calibrated model for long-tailed visual recognition from prior perspective. *Advances in Neural Information Processing Systems*, 34.

[Yang et al., 2021] Yang, Y., Zha, K., Chen, Y.-C., Wang, H., and Katabi, D. (2021). Delving into deep imbalanced regression. *ICML*.

[Zhang et al., 2021] Zhang, Y., Kang, B., Hooi, B., Yan, S., and Feng, J. (2021). Deep long-tailed learning: A survey. *arXiv preprint arXiv:2110.04596*.

[Zhou et al., 2020] Zhou, B., Cui, Q., Wei, X.-S., and Chen, Z.-M. (2020). Bbn: Bilateral-branch network with cumulative learning for long-tailed visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9719–9728.

[Zhu and Zhou, 2021] Zhu, Z. and Zhou, W. (2021). Taming heavy-tailed features by shrinkage. In *International Conference on Artificial Intelligence and Statistics*, pages 3268–3276. PMLR.