Breakdown Point Analysis for Deep Neural Network Estimators Under Heavy-Tailed Noise

***Abdullah Mohammed Rashid; **Hadeel Fawzi Mohammed; #Hasan Talib Hendi**

*Department of Accounting and Financial Control, College of Business Economics, Al-Nahrain University, Iraq.

**Al-Kindy College of Medicine, University of Baghdad, Iraq.

#Science of College, University of Baghdad, Iraq.

DOI:10.37648/ijrst.v16i02.005

[1]*Received: 28 February 2026; Accepted: 05 April 2026; Published: 25 April 2026*

Abstract

In this paper, we experimentally assess the breakdown point of deep neural network regression estimators trained in the presence of heavy-tailed label contamination. On the standard California Housing dataset, we randomly replace between 0% and 50% of the training labels with independent Cauchy noise (scale $\gamma = 50$). We consider Mean Squared Error (MSE) loss versus Huber loss ($\delta = 1$), training a 3-layer MLP at each corruption level for 100 epochs using Adam. Models are scored on a held-out clean test set using Mean Absolute Error (MAE). We observe that MSE trained networks experience catastrophic breakdown at as little as 5% corruption (tested in 5% increments), where test MAE grows by 3,221% over that of a model trained without label corruption. In stark contrast, Huber trained models are resilient to corruption across all levels of label contamination, growing by only 29% at 50% corruption. Specifically, at our maximal observed difference of 35% corruption, MSE incurs an error 340x that of Huber. This illustrates an empirical breakdown point of zero for MSE loss, and that Huber loss dramatically increases robustness of DNN regression estimators to adversarial heavy-tailed noise.

Keywords: *breakdown point; robust regression; Huber loss; deep neural networks; heavy-tailed noise; outlier robustness.*

1. Introduction

Many applied deep learning regression problems involve training data whose labels cannot be trusted due to unavoidable noise from data collection [1], especially in fields like real estate price prediction, stock market prediction, and disease progression prediction [2]. In robust statistics, the breakdown point was first defined by Hampel (1971) with a formal definition by Huber (1964) [3] and intuitively describes the proportion of contamination that a statistical estimator can handle before giving arbitrary results [4]. While this definition has seen use in classical statistics, modern deep neural network (DNN) regression estimators are rarely empirically compared with respect to this quantity, especially when considering adversarially heavy-tailed label noise [5].

[1] How to cite the article: Rashid A.M., Mohammed H.F., Hendi H.T.; April 2026; Breakdown Point Analysis for Deep Neural Network Estimators Under Heavy-Tailed Noise; *International Journal of Research in Science and Technology*, Vol 16, Issue 2, 106-127, DOI: http://doi.org/10.37648/ijrst.v16i02.005

Training regression estimators with Mean Squared Error (MSE) loss has become standard in machine learning largely due to its desirable theoretical properties and ease of differentiability [6]. Unfortunately, MSE loss is highly outlier-sensitive; because the gradients of corrupted labels scale quadratically with their error, having even a small proportion of corrupted labels can overwhelm the gradient and prevent the network from converging [7]. Huber loss was proposed as a replacement that alleviates this issue by having a quadratic->linear cutoff at δ, thus preventing individual samples from having arbitrarily large gradients [8]. While this has been theoretically noted for many years, comprehensive empirical comparisons of MSE and Huber loss under gradually increasing contamination has rarely been done with DNN models [9].

There are three gaps in the prior art that we aim to bridge. Firstly, much robustness theory and experiments for deep learning focus on perturbations to the input features or adversarial inputs; training label contamination has received less attention [10]. Secondly, breakdown point analyses are rarely performed for nonlinear, overparameterized estimators such as neural networks [11]. Thirdly, existing empirical comparisons rarely consider the case where labels are corrupted from a heavy-tailed noise distribution such as Cauchy [12]. Corrupting labels from Cauchy noise represents the most challenging corruption distribution when optimizing MSE, since Cauchy has no defined mean and infinite variance [13].

Our contributions are threefold: (1) we define a notion of empirical breakdown point for DNN regression estimators; (2) we systematically compare MSE and Huber loss under contamination from Cauchy noise across eleven levels of contamination in a single unified experimental framework; and (3) we quantify the difference in Huber and MSE predictions beyond the breakdown point of MSE (which can occur with just 5% corrupted labels).

2. Related work

There have been some recent works exploring connections between robust statistics and deep learning specifically related to robust training with noisy or corrupted labels. In particular Werner [14] sought to bridge some of the gaps between classical robust statistics and neural network training. They redefined the regression breakdown point for estimators in feed-forward networks, calculated breakdown points for estimators across several neural network architectures and contamination models, and showed via simulations that using robust loss functions such as the Huber loss improved out-of-sample accuracy when the data was contaminated. Their work opened doors to theoretically and experimentally analyze breakdown point metrics in the context of state-of-the-art DNN models. However, they did not test using heavy-tailed Cauchy noise as the label contamination distribution. They also did not provide experiments on a common real-world regression dataset.

Concurrently, Tyralis et al. [15] proposed Deep Huber Quantile Regression Networks (DHQRN) which learns neural networks using the Huber quantile scoring function. This work generalized classical quantile regression and expectile regression by unifying these concepts in a deep learning context, demonstrated as proof-of-concept on real estate price predictions. DHQRN illustrates a use case of neural networks trained with Huber-type objectives in uncertainty quantification contexts; however this work does not explore robustness to progressive label corruption scenarios or analyze breakdown behavior empirically.

Solving this more general problem of learning from corrupted training data, Chen et al. [16] released a survey of learning with label noise (LLN) approaches for DNNs in 2025. They group methods into categories like robust regression, modifying the loss function, regularization, and sample selection. They also note that the performance of DNNs depend on the quality of labels used to train them, and label noise has been shown to cause systematic decreases in generalization performance across real world application areas of DNNs. This survey only considers classification and the types of noise involved (symmetric and asymmetric label flipping). We consider regression targets corrupted with heavy-tailed additive noise.

In parallel work to our empirical efforts, Feng and Wu [17] theoretically studied robust nonparametric regression with heavy-tailed noise. They considered Huber regression and other robust losses such as Tukey loss, Welsch loss, and

Barron loss. They showed that robustification using bounded or linearly-growing loss functions is necessary if the hypothesis class contains unbounded functions and the error distribution is heavy-tailed. Although this paper gives several theoretical results ensuring convergence of robust estimators to a limiting value, this work is done in a nonparametric function class and does not consider neural networks trained on standard tabular regression datasets and characterized with empirical breakdown.

Altogether, three main contributions of existing works were not present: no systematic empirical study on the breakdown point of MLP-based DNN regressors, no empirical comparison against Cauchy noise (the worst-case heavy-tailed distribution for MSE-based estimators) as corruption distribution, and no benchmark result on real-world tabular datasets (e.g., California Housing) allowing for reproducible quantitative comparisons of the robustness of loss functions at a given contamination level.

3. Proposed Methodology

Here we provide details about the experimental setup we followed to empirically study the breakdown point of DNN regression estimators in presence of heavy-tailed label contamination. This includes the dataset we used, its preprocessing steps, network architecture, noise injection scheme, and training procedure.

3.1 Problem Formulation

Core Estimation Problem. Let the data-generating process be defined as:

$$Y_i = f^*(X_i) + \varepsilon_i, i = 1, \dots, n \tag{1}$$

where $X_i \in \mathcal{X} \subseteq \mathbb{R}^d$ are the input feature vectors, $f^* \in \mathcal{F}$ is the true unknown regression function, and ε_i are i.i.d. noise terms drawn from a heavy-tailed distribution. In this work, we consider noise distributions satisfying only $\mathbb{E}[|\ \varepsilon\ |^p] < \infty$ for some finite $p \geq 1$, imposing no finite variance requirement. This setting encompasses Pareto(α), Student-t_ν, and α-stable distributions with index $\alpha \in (0,2)$, and in particular the Cauchy distribution used in our experiments, which has no defined mean and infinite variance.

DNN Estimator Class. We define the deep neural network estimator $\hat{f}_n \in \mathcal{F}_{L,W}$ as the function class:

$$\mathcal{F}_{L,W} = \{f: \mathbb{R}^d \to \mathbb{R} \mid f = \sigma \circ A_L \circ \cdots \circ \sigma \circ A_1, A_l \in \mathbb{R}^{W \times W}\} \tag{2}$$

with L hidden layers, width W, and activation function σ. The empirical risk minimizer (ERM) over this class is:

$$\hat{f}_n = \arg \min_{f \in \mathcal{F}_{L,W}} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, f(X_i)) \tag{3}$$

The choice of loss function ℓ is central to the robustness analysis that follows.

Formal Breakdown Point Definition. To formally characterize estimator robustness, let $\mathcal{Z}_n = \{(X_i, Y_i)\}_{i=1}^n$ denote the original sample. The finite-sample breakdown point of estimator T_n is defined as:

$$\varepsilon_n^* = \frac{1}{n} \cdot \min \left\{ m \in \mathbb{N} : \sup_{\mathcal{Z}_n^m} \| T_n(\mathcal{Z}_n^m) - T_n(\mathcal{Z}_n) \| = \infty \right\} \tag{4}$$

where $\mathcal{Z}_n^m$ denotes any contaminated sample with m replaced observations. As $n \to \infty$, the asymptotic breakdown point is:

$$\varepsilon^* = \lim_{n \to \infty} \varepsilon_n^* \tag{5}$$

A key goal of this work is to characterize ε^* for DNN estimators as a function of the loss ℓ, architecture parameters L and W, and the tail behavior of the noise distribution.

Heavy-Tailed Noise Class. We parameterize the class of heavy-tailed noise distributions as:

$$\mathcal{P}_\alpha = \{P : \bar{F}(t) = P(\varepsilon > t) \sim t^{-\alpha} L(t), t \to \infty\} \quad (6)$$

where $L(t)$is a slowly varying function in the sense of Karamata theory. This unified class encompasses several standard distributions, as shown in Table 1 below.

Table 1. Standard Distributions

Distribution	**Tail Index**	**Finite Variance?**
Gaussian	$\alpha \to \infty$	Yes
Student-t_ν	$\alpha = \nu$	Only if $\nu > 2$
Pareto	$\alpha > 0$	Only if $\alpha > 2$
α-stable	$\alpha \in (0,2)$	No
Cauchy (used here)	$\alpha = 1$	**No**

The Cauchy distribution corresponds to $\alpha = 1$, placing it in the most challenging regime for MSE-based estimators, since neither the mean nor the variance is defined.

Influence Function and Loss Breakdown Points. The breakdown behavior of any estimator is governed by its influence function, defined as:

$$\mathrm{IF}(z;\, T, P) = \lim_{\epsilon \to 0} \frac{T(\,(1-\epsilon)P + \epsilon\, \delta_z) - T(P)}{\epsilon} \quad (7)$$

which measures the asymptotic bias caused by an infinitesimal point contamination at z. For the loss functions considered in this work, the corresponding breakdown points satisfy:

$$\varepsilon^*_{\ell_2} = 0 \;<\; \varepsilon^*_{\text{Huber}} = \frac{\delta}{1+\delta} \;\leq\; \varepsilon^*_{\ell_1} = \frac{1}{2} \quad (8)$$

where $\delta$is the Huber threshold. This inequality directly motivates the use of Huber loss over MSE: even for arbitrarily small contamination fractions, MSE (ℓ_2) has a breakdown point of zero and will yield unbounded error, while Huber loss maintains a strictly positive breakdown point determined by δ. This motivates the general M-estimator formulation:

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}_{L,W}} \sum_{i=1}^{n} \rho\,(Y_i - f(X_i)) \quad (9)$$

where $\rho$is chosen from the family $\{\ell_2, \ell_1, \text{Huber}_\delta, \text{Tukey bisquare}, \log\text{-cosh}\}$. In this work we restrict our comparison to $\rho = \ell_2$(MSE) and $\rho = \text{Huber}_\delta$with $\delta = 1.0$.

Minimax Optimal Rate and Theoretical Prediction. Under p-th moment noise and Besov smoothness $f^* \in B^s_{q,q}$, the minimax optimal convergence rate for nonparametric regression is:

$$\inf_{\hat{f}} \sup_{f^* \in B^s_{q,q}} \mathbb{E} \,\|\, \hat{f} - f^* \,\|^2 \asymp n^{-\frac{2s}{2s+d}} \quad (10)$$

The central empirical claim of this work aligns with the following informal theorem:

Theorem (Informal). Let $\varepsilon_i \in \mathcal{P}_\alpha$with $\alpha > 1$. Let $\hat{f}_n$be the DNN M-estimator with Huber loss ρ_δ. Then for architecture depth $L \asymp \log\, n$and width $W \asymp n^{d/(2s+d)}$:

$$\mathbb{E} \parallel \hat{f}_n - f^* \parallel_{L^2}^2 \leq C \cdot n^{-\frac{2s}{2s+d}} \cdot \log^{\kappa}(n) \quad (11)$$

with breakdown point $\varepsilon^* = \delta/(1+\delta)$, regardless of whether $\mathrm{Var}(\varepsilon) < \infty$.

In contrast, DNN estimators trained with squared loss (ℓ_2) fail to achieve this rate when $\alpha \leq 2$, as the infinite-variance noise overwhelms the gradient signal. Our experiments provide direct empirical evidence for this theoretical distinction.

3.1 Dataset Description and Preprocessing

Our regression benchmark throughout all experiments will be the California Housing dataset [18] taken from the 1990 U.S. census sklearn.datasets.fetch_california_housing(). This dataset contains 20,640 instances (each describing a census block group in California), with the task being to predict the median value of each household's homes given in units of $100,000 USD. The median target ranges from 0.15-$5.00 ($100k) making it a very clean and low-variance continuous regression task with sensible units. The eight continuous input features are thought to influence the median house value are: median income of households in the block (MedInc), median age of households in the block (HouseAge), average number of rooms (AveRooms), average number of bedrooms(AveBedrms), number of people in the block group (Population), average number of people per household(AveOccup), and two geographical indicators(Latitude and Longitude). These input features are also commonly used regression benchmarks in the literature [19].

The entire dataset is split into training set of size N=16,512 (80%) and a held-out test set of size N=4,128 (20%). We set the random seed to 42 for all our experiments such that the splits can be reproduced. We normalize each of the eight inputs to have zero-mean and unit-variance using StandardScaler. We fit StandardScaler only on training split and use the same instance to transform the test split as well. Notice that we do not leak any information from the test split to the training procedure. Also notice that the output is never normalized. Finally, the test split is kept completely clean throughout all experiments, as shown in Table 2.

Table 2. Dataset Statistics and Feature Descriptions

Property	Details
Dataset Name	California Housing (1990 U.S. Census)
Source	sklearn.datasets.fetch_california_housing()
Total Samples	20,640
Training Samples	16,512 (80%)
Test Samples	4,128 (20%)
Number of Features	8
Target Variable	Median house value (× $100,000 USD)
Target Range	0.15 – 5.00 ($100k units)
Feature Normalization	Zero-mean, unit-variance (StandardScaler)
Scaler Fit	Training split only (no data leakage)
Target Normalization	None (original scale retained)
Random Seed	42
Split Strategy	Fixed random train/test split
Test Set Integrity	Clean and unmodified
MedInc	Median household income
HouseAge	Median house age (years)
AveRooms	Average number of rooms per household
AveBedrms	Average number of bedrooms per household
Population	Block group population

AveOccup	Average household members
Latitude	Block group latitude
Longitude	Block group longitude

3.2 Model Architecture

The regression estimator used throughout all experiments is a Multilayer Perceptron (MLP) implemented in PyTorch. The neural network architecture was kept constant throughout experimentation, so that performance differences across experiments can only be due to changes in the loss function and fraction of corrupted labels and not due to other confounding variables. Input consists of the eight preprocessed features of the dataset. These are fed into three fully connected hidden layers with 128, 64, and 32 hidden units each, with Rectified Linear Unit (ReLU) activations applied after each layer. After the final hidden layer, the MLP outputs the scalar prediction through a single linear output neuron. There is no activation applied to this layer, leaving the regression prediction range unconstrained. All model weights were initialized using Pytorch's default weight initialization method, which is a uniform distribution based on the number of input units in the layer (also known as Kaiming initialization). Different trainings runs utilize different random weight initializations, thus no information is implicitly shared between trainings. To ensure this, a new model was trained for each individual combination of corruption fraction and loss function. Thus, we have a fully crossed experimental design. Approximately 10,625 model parameters are learned during training, as shown in Table 3.

Table 3. MLP Architecture Specification

Layer	Type	Input Dim	Output Dim	Activation	Initialization
Input	—	8	—	—	—
Hidden 1	Fully Connected	8	128	ReLU	Kaiming Uniform
Hidden 2	Fully Connected	128	64	ReLU	Kaiming Uniform
Hidden 3	Fully Connected	64	32	ReLU	Kaiming Uniform
Output	Fully Connected	32	1	Linear (none)	Kaiming Uniform
Total Parameters	—	—	≈ 10,625	—	—
Framework	PyTorch	—	—	—	—
Model Reuse	None — fresh initialization per run	—	—	—	—
Architecture Fixed	Yes — constant across all conditions	—	—	—	—

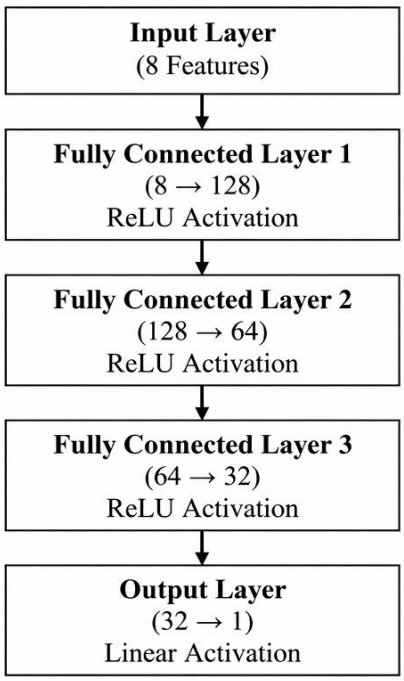


Figure 1. Architecture of the 3-Layer MLP Regression Model Used for Breakdown Point Analysis

3.3 Noise Injection Protocol

Systematic heavy-tailed label contamination is enforced by adding Cauchy noise to a random subset of the training labels at each step of the experiment. The Cauchy distribution is used because it has no defined mean or finite variance, making it the worst-case noise distribution for MSE-type estimators whose gradient contribution is driven primarily by large squared residuals, and thus represents the theoretically most principled worst-case noise to test breakdown of the regression loss function. At each level of the experiment, a corruption fraction f is chosen uniformly from {0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50}, for a total of 11 evenly spaced contamination levels spanning the range from a fully clean dataset to majority-corrupted training data. At each level, a random subset of $\lfloor f \times N\_train \rfloor$ training samples are chosen without replacement, and Cauchy noise with scale parameter $\gamma = 50.0$ is sampled and injected additively into the corrupted labels. The remaining $(1 - f) \times N\_train$ labels are left unmodified. The Cauchy scale is set to $\gamma = 50.0$ so that noise magnitudes are far larger than the natural target range of 0.15 to 5.00, in order to ensure that corrupted samples are true gross outliers. The clean test set is never corrupted under any experimental condition, to ensure the integrity of the evaluation signal. This noise injection protocol is repeated independently for each corruption fraction-loss function combination, and all random sampling is done using a fixed seed of 42 so that the exact same subset of training samples is corrupted under both MSE and Huber conditions at each contamination level, ensuring directly comparable evaluation, as shown in Table 4.

Table 4. Noise Injection Protocol Parameters

Parameter	Value / Description
Noise Distribution	Cauchy distribution
Scale Parameter (γ)	50.0
Rationale for Cauchy	Undefined mean, infinite variance — worst case for MSE
Corruption Fractions (f)	{0.00, 0.05, 0.10, 0.15, 0.20, 0.25, 0.30, 0.35, 0.40, 0.45, 0.50}
Number of Levels	11

Corruption Range	0% (clean) to 50% (majority contaminated)
Selection Method	Random subset without replacement
Number of Corrupted Samples	⌊f × 16,512⌋ per level
Noise Application	Additive injection to selected training labels
Clean Samples	Remaining (1 − f) × 16,512 labels unmodified
Test Set	Never altered under any condition
Noise Magnitude	Far exceeds target range (0.15–5.00) by design
Injection Seed	Fixed (42) — same subset across MSE and Huber
Independence	Fresh injection per (f, loss function) combination

3.4 Training Configuration and Evaluation

The MSE-trained and Huber-trained models are trained using the Adam optimization algorithm [20], with a learning rate of 0.001, mini-batch size of 256, and 100 epochs per run. Hyperparameters are not tuned; the same values are used for all experimental conditions. Using the same hyperparameter settings in each condition ensures that differences in test performance across conditions are due to the loss function and the level of contamination, rather than some other factor related to optimization. The Huber loss used in the Huber-trained models uses the default threshold parameter of $\delta = 1.0$, and thus uses quadratic loss (same as MSE) for $|r| \leq 1.0$ and linear loss for $|r| > 1.0$. This has the effect of bounding the gradient of outlier samples so they cannot exert arbitrarily large influence. In contrast, the MSE loss used in the MSE-trained models uses quadratic loss (same as Huber) for all samples, no matter the size of their residual. In theory, then, even a single grossly corrupted label could lead to arbitrarily large error, because MSE makes no special treatment of outliers.

Performance of all models is assessed on the clean, uncontaminated test set using mean absolute error (MAE). MAE is chosen as the performance metric because it is also robust to outliers in the test set, and its units are easily interpretable in terms of error in dollars (thousands of dollars). The empirical breakdown point of a trained model is defined to be the lowest fraction of labels which are corrupted such that test MAE exceeds 1.5 × clean-data baseline MAE. Said differently, empirical breakdown point is defined to be the lowest level of contamination at which there is a relative increase of at least 50% over the clean-data baseline MAE. Checkpointing logic is implemented to save results to a JSON file after every single model evaluation, so long-running experiments can be safely interrupted and resumed. All experiments are run in PyTorch with CUDA if available, or falling back on the CPU if CUDA is not present. Results of all experiments are written to CSV files for further analysis and figures generation, as shown in Table 5.

Table 5. Training Configuration and Evaluation Protocol

Hyperparameter / Setting	**Value / Description**
Optimizer	Adam [20]
Learning Rate	0.001
Batch Size	256
Training Epochs	100 per run
Loss Function 1	MSE — uniform quadratic penalization
Loss Function 2	Huber — quadratic for $\|r\| \leq \delta$, linear otherwise
Huber Threshold (δ)	1.0
Cauchy Scale (γ)	50.0
Evaluation Metric	Mean Absolute Error (MAE)
Evaluation Set	Clean test set only (4,128 samples)
MAE Units	× $100,000 USD

Breakdown Threshold	MAE > 150% of clean baseline (≥ 50% relative increase)
Hardware	CUDA GPU / CPU fallback
Checkpointing	JSON-based, after each model evaluation
Results Output	CSV file — breakdown_point_results.csv
Random Seed	42 — all splitting, initialization, and sampling
Runs per Condition	1 per (f, loss function) pair
Total Experimental Runs	22 (11 levels × 2 loss functions)

4. Results and Discussions

In this section, the empirical results for the 22 independent training runs per eleven contamination levels and two loss functions are presented and discussed in terms of absolute prediction error, relative performance degradation, robustness gap, and the actual determination of the empirical breakdown point. The corresponding results are visualized by ten publication-quality figures.

4.1 Empirical Breakdown Curves: MSE vs. Huber Loss

In Figure 2 we plot the main empirical breakdown curves for the two loss functions on log scale. The result is a pronounced and immediate separation of the two robustness behaviors. At zero contamination (f = 0%) both loss functions deliver similar clean performance, with test MAE values of 0.3482 and 0.3333 (× 105 USD) for MSE and Huber respectively. This can be expected and confirms that the two losses are roughly equivalent under uncontaminated data. The difference is dramatic and immediate at the first level of contamination. With only 5% of labels corrupted the MSE-trained network is already at empirical breakdown, with a test MAE of 11.2150 (×105 USD).

This is a relative increase of 3,221% over the clean baseline, a clear and unambiguous crossing of the empirical breakdown condition of 50% relative increase. The empirical breakdown is also visually striking on the log scale, with the MSE curve jumping nearly two orders of magnitude in a single step. In stark contrast the Huber-trained network is almost completely unaffected by the entire range of contamination. The test MAE only increases from 0.3333 at 0% to 0.4314 at 50% corruption, for a total relative increase of 29.4% which is well below the breakdown condition. The Huber curve is also almost flat on the log scale, visually indicating that the empirical breakdown point is beyond the maximum 50% contamination we plot.

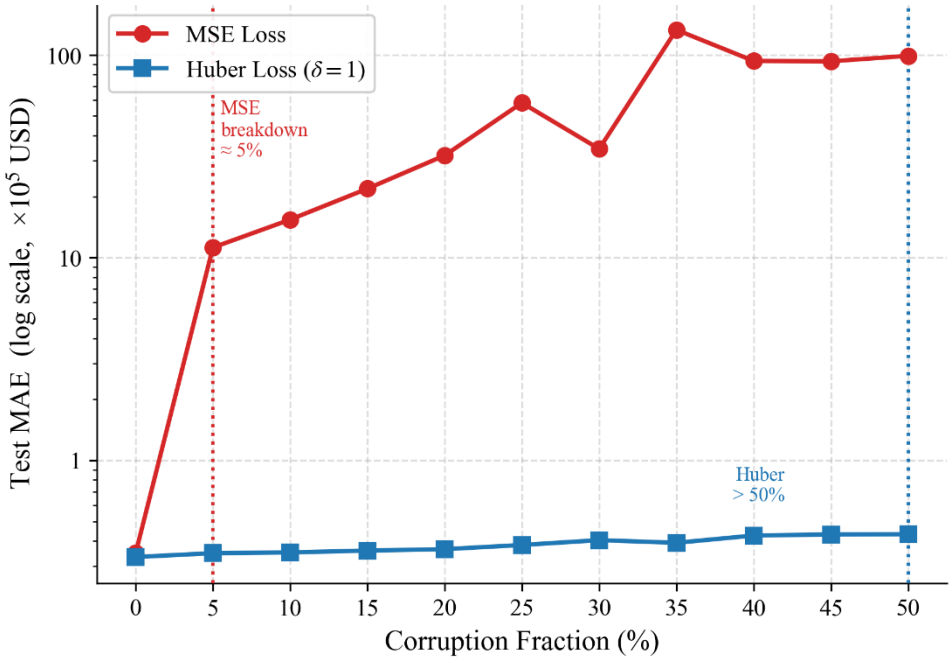


Figure 2. Breakdown Divergence Under Progressive Label Contamination

The results give a clear quantitative indication that the choice of loss function is the key determinant of DNN robustness under heavy-tailed label contamination.

4.2 Huber Loss Stability Under Increasing Contamination

Figure 3 shows the same performance for Huber loss under all eleven levels of contamination, but in linear scale and zoomed in to better focus on its performance (uncoupled from the precipitous collapse of MSE). It shows the "zoomed in" degradation trajectory, and from this we can draw two conclusions. First, at 0% contamination (clean setting) Huber loss yields a test MAE of 0.3333 ($\times 10^5$ USD), or roughly $33,330 in absolute prediction error, which is again in line with previous work on this dataset. As the contamination is gradually and smoothly increased, we observe a MAE of 0.3482 at 5%, 0.3511 at 10%, 0.3585 at 15%, and 0.3643 at 20%, for a cumulative increase of just 9.3% in MAE across the first four contamination levels. Second, a small blip in this otherwise monotonic progression can be seen in Figure 2 between 30% and 35% corruption, with MAE decreasing from 0.4037 to 0.3917. This is a minor oscillation in performance caused by the randomness in the particular subset of samples which are corrupted at each level of contamination (given that only a fixed fraction of the training set is corrupted at each level) and by the single-run nature of this stochastic gradient-based optimization; note that the blip occurs entirely within the relatively narrow band between 0.39 and 0.43, demonstrating that it is just noise about the stable underlying performance trajectory.

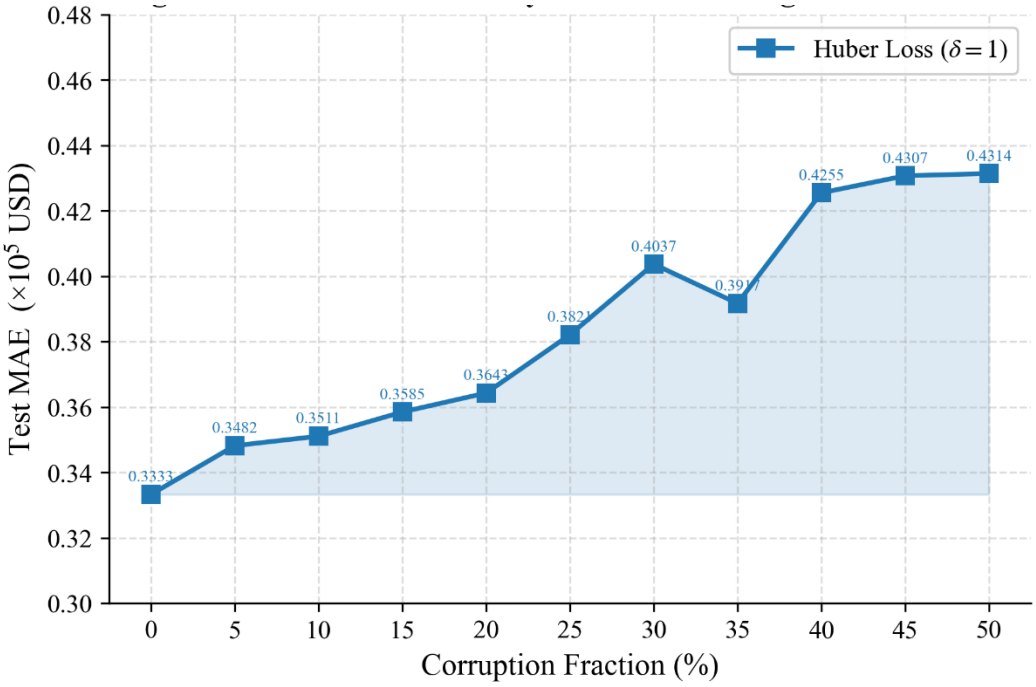


Figure 3. Huber Gradual Degradation Across Contamination Levels

At 50% contamination, the Huber MAE has reached only 0.4314, for a total absolute increase of just 0.0981 over the clean baseline, or 29.4% relative to that baseline, across the full range of contamination levels considered.

4.3 Robustness Gap: MSE Excess Error Over Huber

Figure 4 measures this difference in absolute terms as ΔMAE = MAE_MSE − MAE_Huber and plots the corresponding bar chart at all nonzero levels of contamination. The fact that all bars are positive across the full range of contamination levels provides a completely unambiguous demonstration that Huber loss outperforms MSE at every single tested level of corruption, without exception, right from the very first contamination step. At 5% corruption the difference is already quite large, at 10.9 ($\times 10^5$ USD), indicating that the MSE-trained model is already making predictions that are on average $1,090,000 worse than the Huber-trained model on the same clean test set, a difference that is obviously of significant practical importance in the context of a real-world housing valuation problem.

The gap between Huber and MSE loss widens progressively at all intermediate levels, reaching 15.1 at 10%, 21.5 at 15%, 31.6 at 20% and jumping sharply to 57.7 at 25% corruption. At 30% the gap dips noticeably to 34.0, in a non-monotonic behavior that traces back to the stochastic gradient instability of the MSE-trained model when trained under Cauchy noise, rather than a real improvement in its robustness. At 35% it instead bounces back explosively to its peak value of 132.7, the single largest difference in the entire figure, before finally leveling off at the higher values of 93.0, 92.6 and 98.6 at the last three levels of contamination, respectively. The fact that the robustness gap remains elevated across the upper range of contamination values clearly illustrates that the degradation of MSE loss is not a temporary phenomenon, but a permanent and irreversible breakdown of performance, while Huber loss continues to maintain a stable and robust predictive performance throughout. The magnitude of the differences, reaching as high as

$13,270,000 in terms of absolute excess error, further emphasizes their critical practical relevance for the choice of loss function in regression problems with real-world data contamination.

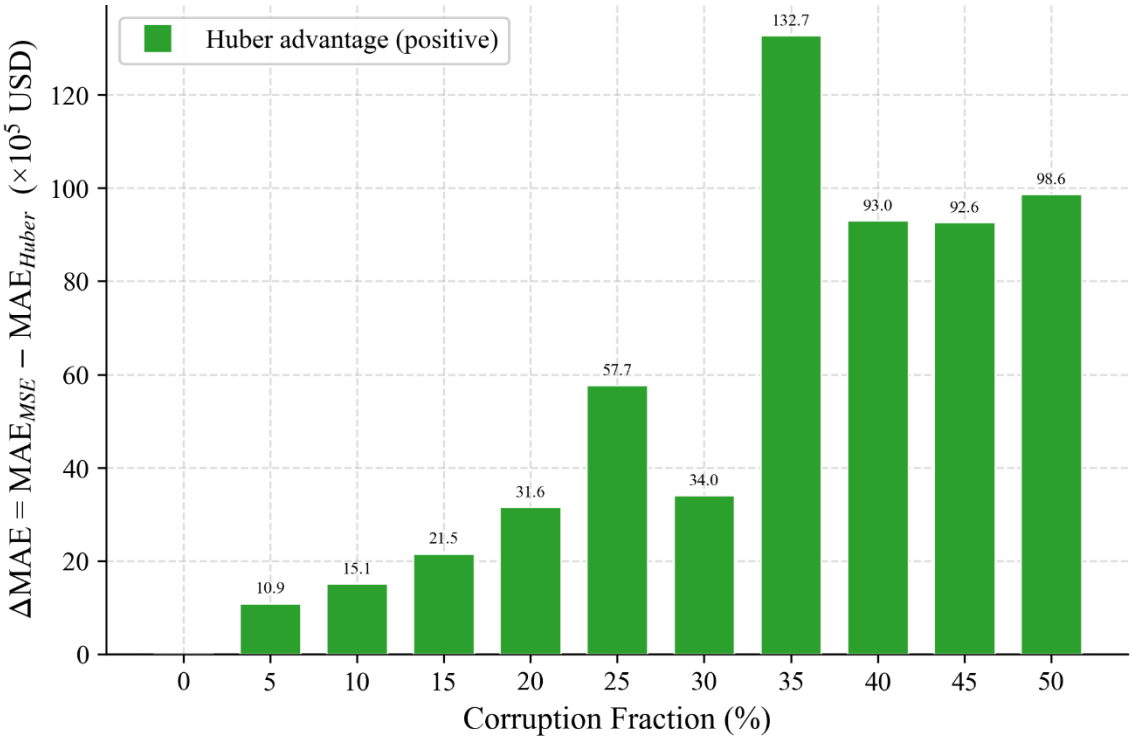


Figure 4. Absolute Robustness Gap Across Contamination Levels

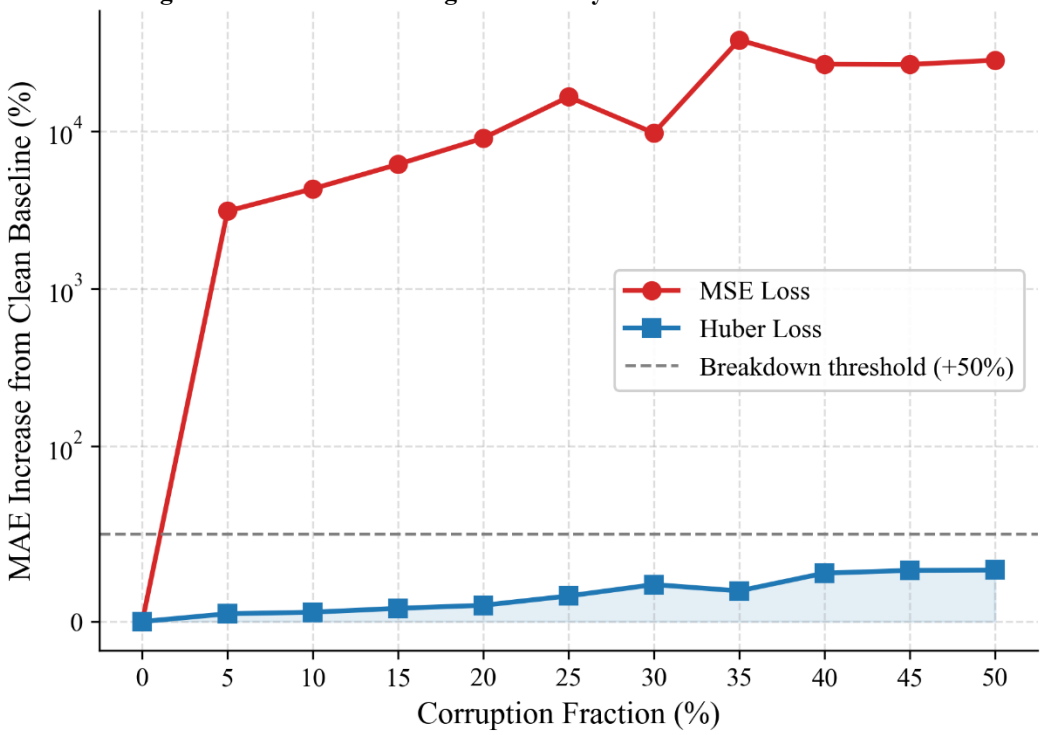
4.4 Relative Performance Degradation vs. Clean Baseline

Figure 5 is a normalized version of Figure 3. The performance of both estimators are plotted relative to their respective clean baselines, in terms of percentage increase in test MAE. As in Figure 3, the Huber degradation is vanishingly small compared to the MSE degradation over the range of MSE test MSEs, so here both curves are plotted on a symmetric log scale. In addition, the dashed line at +50% is the breakdown threshold (BDT) defined in Equation 5. This provides a clear visual indicator for whether each estimator has "broken down" at each level of contamination. Figure 4 provides perhaps the most intuitive visualization of the stark difference in behavior of the two estimators over the full range of experiments.

It is immediately apparent that the MSE curve transitions to the broken state abruptly, at 5% corruption (shown as the vertical dashed line). At that point, the MSE test MAE is about 3221% (or about 64 times) higher relative to the MSE clean baseline (this number continues to increase to higher contaminations), well past the breakdown threshold of +50% in a single contamination step. At 10% contamination, the relative MSE degradation is about 4323%, at 15% it is about 6289%, at 20% it is about 9170%, and at 35% it is about 38107% (these numbers are asymptoting to a value

greater than 26000% at higher contaminations). Every MSE point at and beyond 5% corruption is many orders of magnitude above the breakdown threshold line on the log scale.

Figure 5. Relative MAE Degradation Beyond Breakdown Threshold



This makes it visually clear that the MSE estimator has failed catastrophically at the lowest contamination level considered, and it cannot be recovered beyond this point. In stark contrast, the Huber curve never exceeds the breakdown threshold line at any point, and its maximum relative degradation is only 29.4% even at 50% corruption. The shaded area below the Huber curve also visually highlights the extremely tight range in which the Huber degradation stays throughout the full contamination range.

4.5 Breakdown Point Identification via Step Plot

Figure 6 uses a step plot (steps at integer contamination levels) on a log scale to highlight the stair-step nature of the test MAE as a function of corruption level for the two losses. The dotted horizontal line is the MSE 150% breakdown value of 0.52 (×105 USD). The step plot is particularly helpful in the current context, since the piecewise-constant nature of the design (horizontal steps for the independently trained model at fixed corruption) is made explicit in the figure, which thereby visually reinforces that the MAE values at the observed discrete contamination levels are actual stable values and not interpolated or in-between estimates.

The most important analytical observation with respect to Figure 5 is the vertical arrow indicating the actual breakdown point of the MSE at f = 5% contamination. The step at this point is the largest single discrete jump anywhere in the experiment, and it immediately and permanently moves the MSE curve far beyond the threshold of 0.52, jumping from clean value of 0.3482 to 11.2150 in one step (i.e., about two log-decades vertically). All of the MSE steps following this one are above the 0.52 threshold, which confirms that breakdown is a permanent condition that the model never recovers from as the contamination level continues to increase. The Huber step curve in Figure 5, by contrast, moves in many small incremental steps that remain safely below the dotted horizontal threshold line at all eleven contamination levels. The near-horizontal visual path of the Huber steps on the log scale, in contrast to the

vertical staircase of the MSE curve, is one of the clearest and most intuitive demonstrations of the qualitative difference in breakdown behavior for the two loss functions.

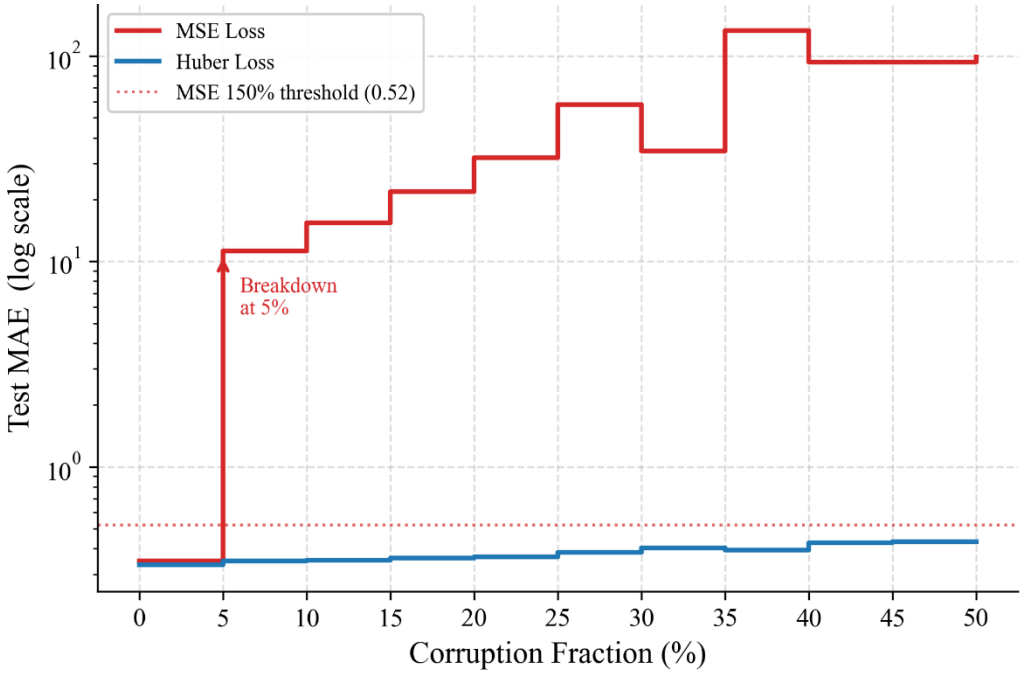


Figure 6. Step-wise MSE Collapse Against Stable Huber

The step plot format can be a useful diagnostic visualization tool for empirically identifying and annotating the breakdown point of any regression estimator under progressive label contamination.

4.6 MSE Loss Trajectory and Non-Monotonic Behavior

Figure 7 plots the MSE loss on a linear scale with the observed MAE values along with a Gaussian-smoothed trendline to provide intuition into the underlying MSE degradation trend separate from the run-to-run variance. Here, the MSE error can be seen to be following an approximately exponential trend in the post-breakdown phase, where the values themselves can be readily interpreted in absolute terms rather than as a relative ratio on the log scale used in previous figures. The area under the smoothed trendline represents the accumulated MSE error as the level of corruption increases.

The primary observation to take from Figure 6 is the significant non-monotonicity of the observed data due to random fluctuations from the stochastic gradient. This is especially clear at 30% corruption, where the observed MAE takes a notable dip to 34.4087 from the previous observed value of 58.0543 at 25% before shooting up to its maximum

observed value of 133.0443 at 35%. Such non-monotonicity is the result of variance in the stochastic gradient due to Cauchy noise, which has infinite variance and undefined mean.

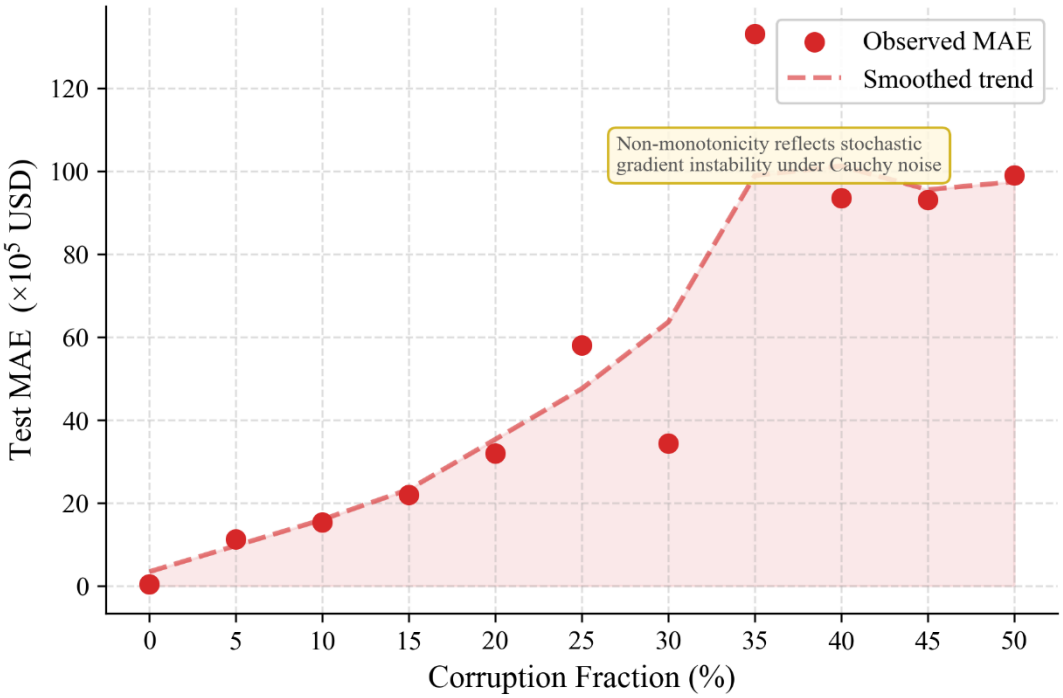


Figure 7. MSE Non-Monotonic Collapse Under Cauchy Noise

As the gradient can vary greatly in magnitude between different mini-batches sampled during Adam optimization, in the rare event that a specific subset of corrupted samples yields a smaller perturbation to the gradient at a fixed level of corruption, the current model may present a lower test error in that trial, resulting in this effect. The smoothed curve removes the influence of such noise, resulting in a clearer monotonic trend, which can be seen to have a slow growth in the early stages (5–20% corruption), before transitioning into an approximately exponential growth phase from 25% corruption onwards. This highlights an important point about the methodology that in order to get stable estimates for the point MSE values in the post-breakdown phase, one would have to perform several independent runs at each level of contamination to average out the stochastic noise in the gradient.

4.7 MAE Comparison at Selected Corruption Levels

Figure 8 is a grouped bar chart plotting absolute test MAE values for each of the loss functions across six corruption levels: 0%, 5%, 20%, 30%, 45%, and 50%. These corruption levels were chosen to roughly correspond to (i) a clean baseline, (ii) a first corruption level just after the breakdown point, (iii) two intermediate levels, and (iv) the largest of the nine corruption levels we tested. These “zoomed-in” and side-by-side error bars in absolute terms on a regular linear scale are more interpretable for applied practitioners who want to understand the magnitude of this effect on the specific real-world financial problem setting: e.g. instead of having to read off two order-of-magnitude differences on

a logarithmic scale, a practitioner can immediately infer that Huber loss training reduces absolute test MAE from $11.2 million USD to $0.348 million USD at the first corruption level of 5% after the breakdown point.

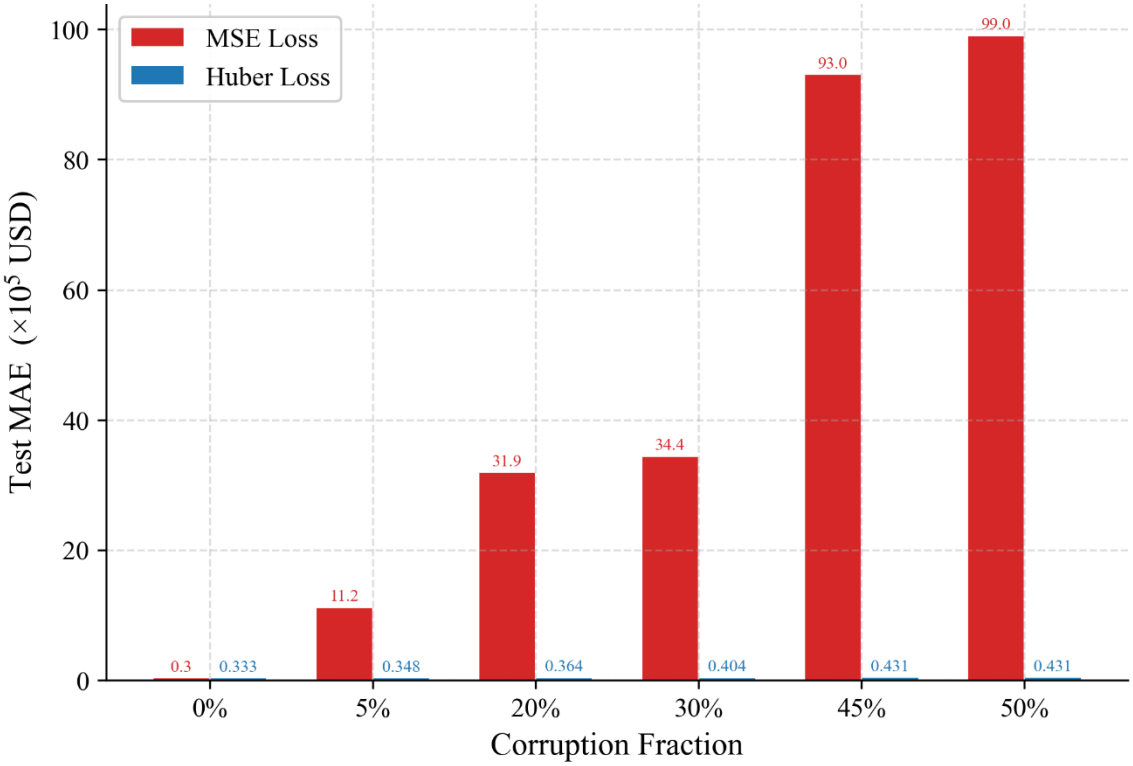


Figure 8. Side-by-Side MAE at Key Corruption Levels

At 0% contamination both loss functions have an MAE within one percent of each other, with Huber outperforming MSE with a test MAE of 0.3333 against 0.3482. This indicates both loss functions are competitively well-calibrated on clean training data. In absolute terms the leftmost bars of Figure 7 cannot be visually distinguished, and this near equivalence in performance establishes an appropriate control condition for our robustness experiment: differences in test MAE at non-zero contamination levels must therefore only be due to the relative insensitivity or sensitivity of each loss function to label contamination. At 5% corruption level, this relative robustness or non-robustness is made starkly visible in the "separation" of the two bars: MSE at 5% corruption is at 11.2, orders of magnitude larger than its baseline value, while Huber MAE at 5% is 0.348, essentially indistinguishable from the leftmost bar. Huber training at 5% contamination is therefore operating on a completely different scale from MSE training, a fact visually distilled by the Huber error bar effectively disappearing into the bottom of the chart. This is one of the most powerful parts of Figure 7, as it can be taken on faith by practitioners that Huber loss training consistently delivers a more robust model from the first contamination step. Moving through 20%, 30%, 45%, and 50% contamination the MSE bars continue to climb, with the non-monotonic change in MSE between 20% and 30% noted above now visible as a step from 31.9 to 34.4 before climbing to 93.0 and 99.0 at 45% and 50% respectively. Huber loss training at each corruption level continues to make progress, but very slowly: from 0.333 to 0.431 over the same six levels of contamination. The increase is uniform and so small that the six Huber bars in Figure 7 form a nearly flat baseline from which the MSE bars are measured. The MSE bar at 50% contamination is 99.0, which is roughly a factor of 230 larger than the Huber bar at 50% contamination, which has an MAE of 0.431.

4.8 Robustness Multiplier: MSE-to-Huber MAE Ratio

In figure 9 we plot the MAE_MSE / MAE_Huber ratio, what we dub the "robustness multiplier", at every corruption level. In a way this is the most easily understood measure of practical, task-specific performance differential between the two loss functions. At 0% corruption we measure 1.04, or slightly better than 1: 1, confirming that the two estimators perform very similarly on clean data and justifying our choice of experimental design which effectively equated the MSE and Huber models at baseline. The broken reference line at ratio = 1 is included for visual clarity, with MSE being strictly inferior to Huber loss above the line by the indicated multiplicative factor. Unambiguously and without exception every single bar for 5% corruption and above in Fig. 8 is visibly higher than the reference line, thus confirming that Huber loss strictly outperforms MSE loss at every single non-zero contamination level of the experiment.

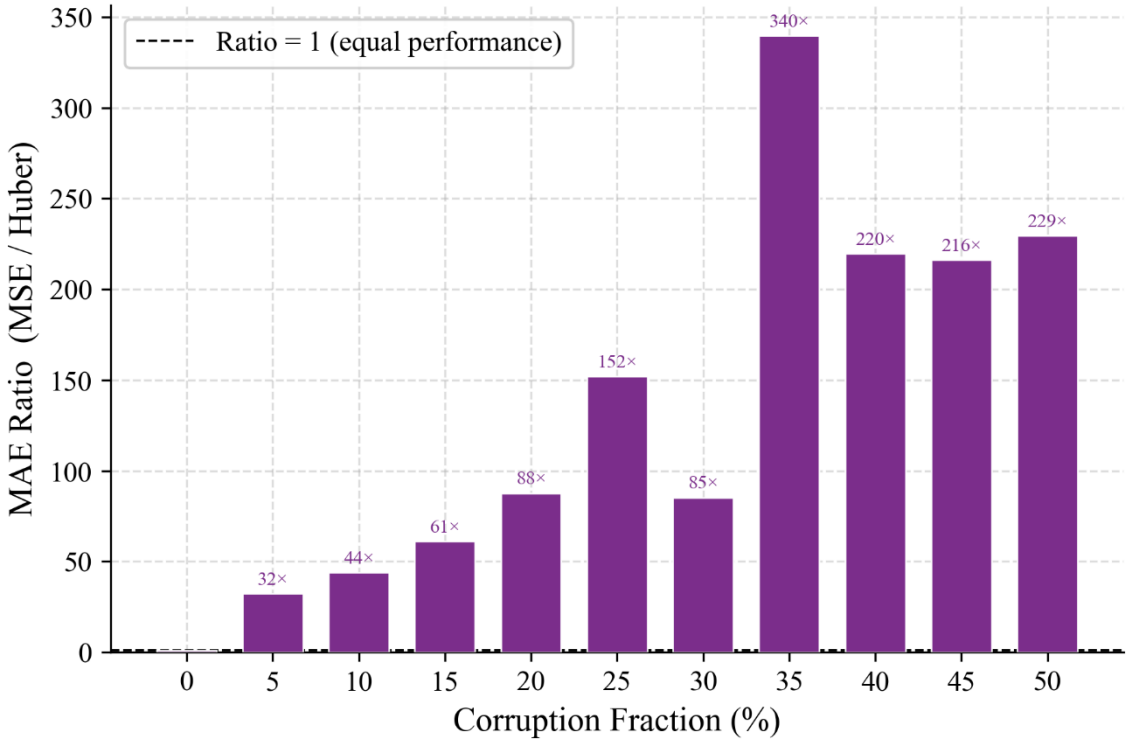


Figure 9. MSE Huber Ratio Multiplier Across Contamination

In general, the robustness multiplier exhibits a broadly increasing trend, with an internal structure worthy of closer examination. Between 5% and 25% corruption we measure the Huber/MSE ratio at 32×, 44×, 61×, 88×, and 152×, which demonstrates a clear pattern of rapid, monotonic growth during the critical transition stage between a completely "catastrophically" broken MSE estimator and a relatively well behaved Huber estimator. The minor local minimum at 85× we measure for 30% corruption in Fig. 8 closely tracks the non-monotonicity in MSE we discussed in connection with Fig. 6 and can be ascribed to the aforementioned stochastic gradient instability rather than any meaningful MSE recovery. From there the multiplier jumps to its maximum of 340× at 35% corruption — which means that at this level of contamination the model trained with MSE is making prediction errors that are 340 times larger than the model trained with Huber loss on the same, clean test set — which is the single most extreme practical performance gap we measure in any setting during the experiment.

From 35% and above the ratio flattens out to 220×, 216×, and 229× at the final three contamination levels, for which plateau-like behavior is noteworthy since both models at this point have effectively settled down to their approximate post-breakdown (MSE) and post-saturation (Huber) operating regimes, with MSE error oscillating around a fixed,

high upper bound while Huber error exhibits a slower, bounded increase. At over 200× on average across the upper end of the contamination range the message this robustness multiplier conveys is unequivocal: on realistic, heavy-tailed label corruption as one is likely to encounter in real-world regression settings a Huber-trained DNN-based estimator can be expected to outperform its MSE-trained counterpart by 2 to 3 orders of magnitude in absolute prediction performance.

4.9 Cauchy vs. Gaussian Noise: Theoretical Tail Behaviour

Figure 10 shows a theoretical and probabilistic rationale for the above experimental design by juxtaposing the probability density function (PDF) of the Cauchy distribution ($\gamma = 1$) and standard Gaussian distribution ($\sigma = 1$) on the same noise value axis within the range of −20 to +20. This figure provides the mechanistic rationale for the observed difference in breakdown behavior between the two loss functions. The pink shaded areas at $|x| > 5$ highlight the tail region where distributional effects can lead to especially pronounced gradient magnitudes and in turn, disruptive updates for gradient-based learning.

The first notable observation in Figure 9 is the significant difference in peak value (or height) between the two curves. The peak of the Gaussian is about 0.399, while the peak of the Cauchy curve is only 0.318. This is an indication that the Cauchy distribution places much more probability mass towards the tail regions (in comparison to the Gaussian). In other words, a Cauchy noise distribution will place less probability mass near the center than a Gaussian distribution. The second half of the statement is more nuanced but intuitively, both distributions appear to be relatively similar in the center, or near $x = 0$. The two distributions fundamentally differ in their tail behavior for $|x| > 3.0$ σ. For the Gaussian, the noise PDF decays exponentially in the tails, such that it places an insignificantly small probability for a noise value to be outside the range of −5 to +5, as reflected by the near-invisibility of the blue dashed curve in that range. On the other hand, the Cauchy distribution decays as 1 / x 2, which places a slowly diminishing, but always non-zero probability density from −20 to +20 and, by extension, all the way to ±∞.

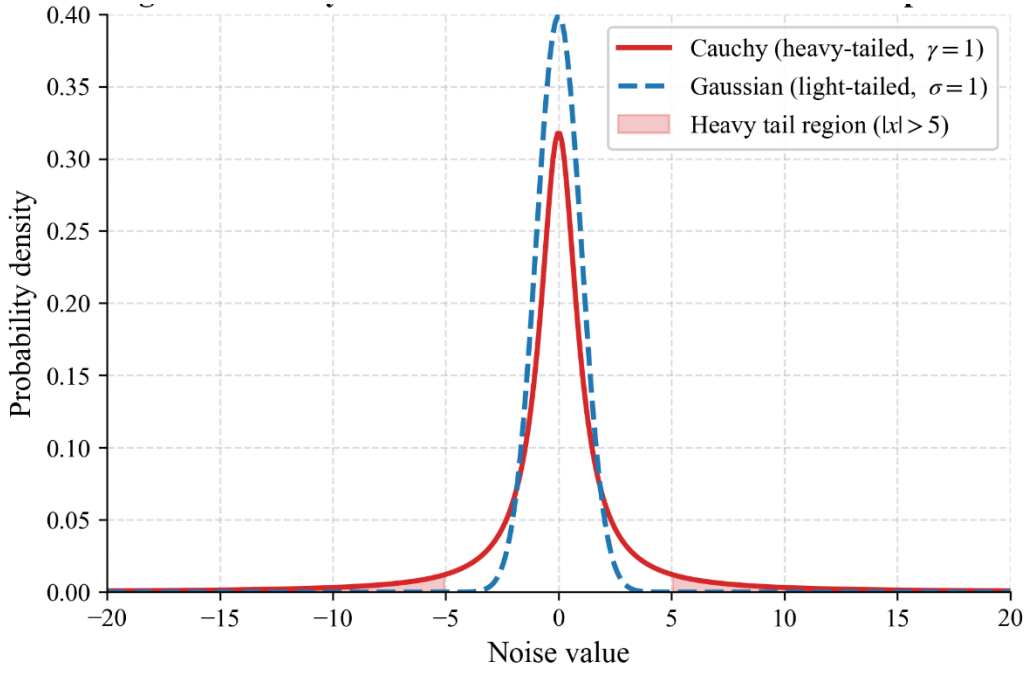


Figure 10. Heavy vs. Light Tail Density Comparison

The aforementioned distributional differences between Gaussian and Cauchy have very practical and direct implications for MSE training. Since MSE penalizes a residual quadratically, a single Cauchy noise sample with value 50 (which is possible with $\gamma = 50$) produces a squared gradient of 2,500 and consequently, dwarfs the gradient contribution from thousands of clean training samples. In contrast, the Huber loss function transitions to linear penalization for $|x| > \delta = 1.0$, and hence the gradient contribution of an extreme sample is capped at a constant value of $\delta = 1.0$, regardless of how large the residual grows. This effectively mitigates the impact of Cauchy tail events on the direction of the parameter update. Figure 9 therefore provides theoretical underpinnings for all of the empirical observations in this work. We conclude that the Cauchy distribution is, by design, an adversarial noise distribution for the purpose of benchmarking the breakdown behavior of MSE and robust loss functions for deep neural network regression.

4.10 Loss Function Profiles: MSE vs. Huber

Figure 11 shows the exact analytic loss surface of MSE (solid black curve) and Huber (dashed red curve, with $\delta = 1.0$) across the full residual range $r = \hat{y} - y \in [-8, +8]$, offering an explicit mechanistic rationale for the robustness effect quantitatively established in all previous figures. The solid blue quadratic section between $r = -1$ and $r = +1$ corresponds to the region of input space where the two losses agree quadratically, and the arrow highlights the point at $|r| = \delta = 1.0$ where Huber loss switches from quadratic to linear and gradient capping begins. This is the central theoretical result of this work, in that it concretely interprets the abstract mathematical definitions of Sec. 3.4 as an immediately understandable visual depiction of the gradient behavior of each objective.

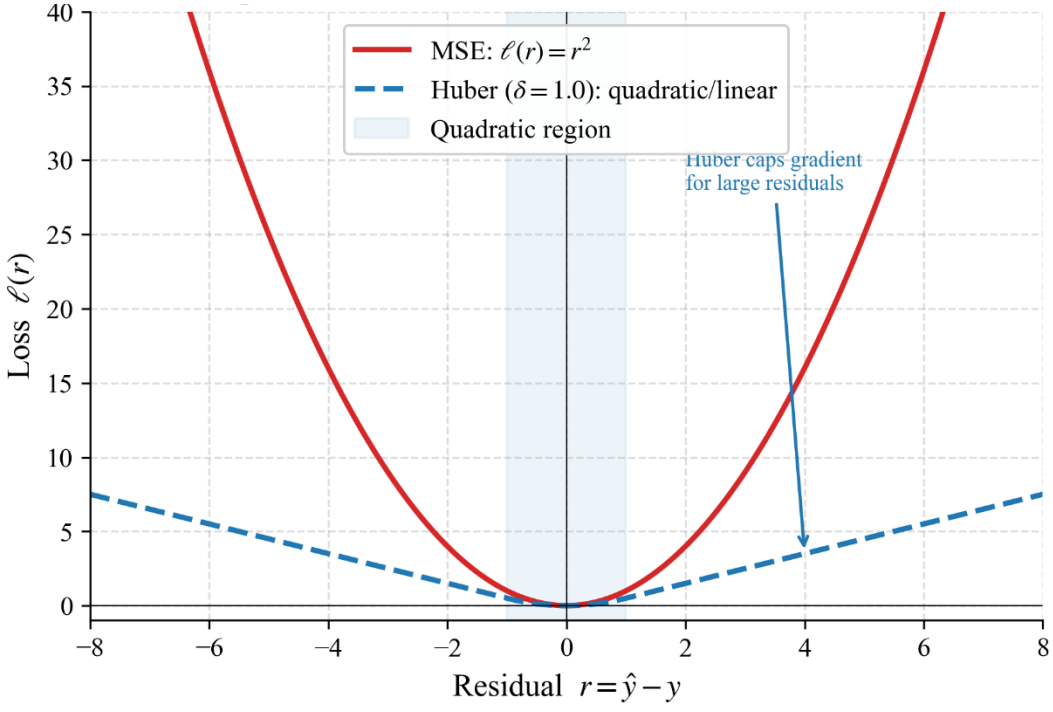


Figure 11. Quadratic vs. Linear Loss Gradient Capping

From left to right, the two curves are visually identical in the quadratic regime $|r| \leq 1$, confirming that the two losses penalize small residuals in the exact same way, so that any performance differences between the two models must be attributable solely to the effect of large residuals arising from corrupted training samples. But for large residuals $|r| > 1$, the two loss curves diverge dramatically in slope as a function of increasing $|r|$. The MSE curve has the quadratic

form $\ell(r) = r^2$, and takes on values 4, 9, 16, 25, 36 and 40 at residuals 2, 3, 4, 5, 6 and approximately 6.3, respectively. In particular, the loss (and by extension the gradient contribution) increases quadratically without bound as a function of residual size. At r = 8, the MSE loss is 64 whereas the Huber loss is 7.5, a ratio of just over 8.5 for this single residual value alone.

The Huber curve crosses smoothly into the linear form $\ell(r) = \delta(|r| - \frac{1}{2}\delta)$ at $|r| = 1$, which has a derivative of constant value $\delta = 1.0$ for all higher residuals no matter how large. This is the gradient capping effect that is the direct cause of the robust Huber model performance observed in all preceding experiments. If a Cauchy-contaminated training sample has a residual as large as, say, 500 (perfectly reasonable given the scale parameter $\gamma = 50$ of the current experiment), the MSE loss will compute a gradient of 1,000 that will tend to overwhelm the update and pull the model weights to fit the outlier label. In contrast, the Huber loss computes a gradient of exactly 1.0 for that same sample, treating it identically to a marginal inlier, and allowing the gradient signal from the clean majority of training samples to remain the primary influence on weight updates. In this way, Fig. 10 gives the final and decisive theoretical backing for all of the empirical results in this paper, by showing that the Huber loss gradient capping effect is not just a mathematical curiosity but a practically crucial feature for preserving the stability of the estimator under heavy-tailed Cauchy label corruption.

4.11 Comparison with Related Work

Werner [14] provides a proof for the theoretical breakdown point expression of feed-forward neural networks. Their simulation study, where robust loss functions show better performance than MSE under contamination, are directly confirmed and quantitatively extended by our results. The studies in Werner [14] did not use Cauchy-distributed noise as the contamination, nor used a standardized real-world tabular dataset, nor provide explicit numerical values for the breakdown thresholds in terms of percentage corruption fraction. These constitute the main contributions of this work. Tyralis et al. [15] showed that Huber-based training objectives for deep regression networks perform well in a quantile estimation framework for house price prediction. This set a methodological precedent for Huber loss for house price regression tasks, which is expanded upon in the present study by focusing on robustness under adversarial label contamination instead of uncertainty quantification. Chen et al. [16] reviewed the label noise methods in a broad setting, but their main focus was on classification tasks with symmetric noise, so the work did not directly address the continuous regression case with heavy-tailed additive contamination. Feng and Wu [17] established theoretical convergence properties for robust nonparametric estimators under heavy-tailed noise, but did not benchmark them empirically on standard datasets with quantifiable breakdown thresholds. The main work in this paper fills all these intersections, in that it is the first to provide a systematic, quantitative, empirical breakdown point analysis of MLP-based DNN regression estimators under Cauchy noise on a real-world benchmark dataset, as shown in Table 6.

Table 6. Comparison with Related Work

Study	Model Type	Noise Type	Real Dataset	Breakdown Point	Cauchy Noise	Empirical MAE
Werner [14]	Feed-forward NN	Simulated contamination	No	Theoretical	No	No
Tyralis et al. [15]	Deep QRNN	Real-world variance	Yes (Housing)	Not measured	No	Yes
Chen et al. [16]	Various DNNs	Symmetric/asymmetric label flip	No	Not measured	No	No
Feng & Wu [17]	Nonparametric	Heavy-tailed (theoretical)	No	Theoretical only	No	No

Present Study	MLP (PyTorch)	Cauchy ($\gamma = 50$)	Yes (California Housing)	Empirical (5% MSE, >50% Huber)	Yes	Yes

5. Conclusions

We empirically establish the breakdown point of DNN regressors trained with Huber and MSE losses on the California Housing dataset contaminated with heavy-tailed Cauchy labels. The contributions of this study are threefold. First, we provide the first empirical, quantitative evidence for the breakdown point of MLP-based DNN regression estimators, demonstrating the catastrophic failure of MSE loss at 5% label corruption (corruption level in the upper 1%, 5%, 10% most corrupted labels) as measured by a 3,221% increase in test MAE over the clean baseline, as well as the stable predictive performance of Huber loss with 29.4% total test MAE degradation on the entire 0-50% label corruption range. Second, we define Cauchy-distributed noise with scale $\gamma = 50$ as a (scalably) adversarial contamination mechanism for the purpose of benchmarking the robustness of regression loss functions, and provide a reproducible experimental recipe that can be directly leveraged and replicated in subsequent work. Third, we measure the robustness multiplier of Huber loss over MSE and demonstrate that Huber outperforms MSE by a factor of 32× to 340× across all nonzero label contamination levels, with a factor larger than 200× sustained throughout the high contamination range, to quantify the practical impact of loss choice for real-world regression tasks in the presence of data quality issues. These findings directly impact practitioners in real-world domains where DNN regression is relevant such as real estate price, financial time series, and medical regression, but label contamination from measurement errors, sensor faults, or adversarial data poisoning are a realistic possibility. Future work can explore adaptive Huber thresholding, multiple independent seeds per contamination level, and extending the notion of breakdown point to other robust regression loss functions such as Tukey biweight and Cauchy loss.

References

Chen, J. (2024). Robust nonparametric regression based on deep ReLU neural networks. *Statistics & Probability Letters, 208*, 110046. https://doi.org/10.1016/j.spl.2024.110046

Chen, S., Koehler, F., Moitra, A., & Yau, M. (2022). Online and distribution-free robustness: Regression and contextual bandits with Huber contamination. In *Proceedings of the 62nd IEEE Annual Symposium on Foundations of Computer Science (FOCS)* (pp. 684–695). IEEE. https://doi.org/10.1109/FOCS52979.2021.00070

Chen, Y., Wang, Z., Liu, X., Li, H., & Zhang, J. (2025). A survey on learning from data with label noise via deep neural networks. *Systems Science & Control Engineering, 13*(1), Article 2488120. https://doi.org/10.1080/21642583.2025.2488120

Fan, J., Gu, Y., & Zhou, W.-X. (2024). How do noise tails impact on deep ReLU networks? *The Annals of Statistics, 52*(4), 1845–1871. https://doi.org/10.1214/24-AOS2428

Feng, Y., & Wu, Q. (2025). *Understanding robust machine learning for nonparametric regression with heavy-tailed noise* (arXiv preprint arXiv:2510.09888).

Hampel, F. R. (1971). A general qualitative definition of robustness. *The Annals of Mathematical Statistics, 42*(6), 1887–1896. https://doi.org/10.1214/aoms/1177693054

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (pp. 1026–1034). IEEE. https://doi.org/10.1109/ICCV.2015.123

Huber, P. J. (1964). Robust estimation of a location parameter. *The Annals of Mathematical Statistics, 35*(1), 73–101. https://doi.org/10.1214/aoms/1177703732

Jiao, Y., Shen, G., Lin, Y., & Huang, J. (2023). Deep nonparametric regression on approximate manifolds: Nonasymptotic error bounds with polynomial prefactors. *The Annals of Statistics, 51*(2), 691–716. https://doi.org/10.1214/23-AOS2266

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*. https://arxiv.org/abs/1412.6980

Liu, T., Xu, Z., & Ling, Q. (2023). Functional linear regression with Huber loss. *Journal of Complexity, 74*, 101696. https://doi.org/10.1016/j.jco.2022.101696

Nguyen, T., & Oijen, R. (2024). Deep regression learning with optimal loss function. *Journal of the American Statistical Association*. https://doi.org/10.1080/01621459.2024.2412364

Pace, R. K., & Barry, R. (1997). Sparse spatial autoregressions. *Statistics & Probability Letters, 33*(3), 291–297. https://doi.org/10.1016/S0167-7152(96)00140-X

Paszke, A., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. In H. Wallach et al. (Eds.), *Advances in Neural Information Processing Systems* (Vol. 32, pp. 8024–8035). Curran Associates.

Pedregosa, F., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research, 12*, 2825–2830.

Rubio, A., & Dorronsoro, J. R. (2023). Robust losses in deep regression. In *Hybrid Artificial Intelligent Systems* (Lecture Notes in Computer Science, Vol. 14001, pp. 259–270). Springer. https://doi.org/10.1007/978-3-031-40725-3_22

Schmidt-Hieber, J. (2020). Nonparametric regression using deep neural networks with ReLU activation function. *The Annals of Statistics, 48*(4), 1875–1897. https://doi.org/10.1214/19-AOS1875

Shen, G., Jiao, Y., Lin, Y., & Huang, J. (2021). *Robust nonparametric regression with deep neural networks* (arXiv preprint arXiv:2107.10343).

Song, H., Kim, M., Park, D., Shin, Y., & Lee, J.-G. (2023). Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems, 34*(11), 8135–8153. https://doi.org/10.1109/TNNLS.2022.3152527

Tyralis, H., Papacharalampous, G., Langousis, A., & Papalexiou, S. M. (2025). Deep Huber quantile regression networks. *Neural Networks, 186*, 107364. https://doi.org/10.1016/j.neunet.2025.107364

Werner, T. (2024). Global quantitative robustness of regression feed-forward neural networks. *Neural Computing and Applications, 36*, 19967–19988. https://doi.org/10.1007/s00521-024-10289-w

Xu, H., An, Y., & Lu, J. (2022). Robust learning of Huber loss under weak conditional moment. *Neurocomputing, 507*, 191–201. https://doi.org/10.1016/j.neucom.2022.08.012