AI for Statistical Analysis

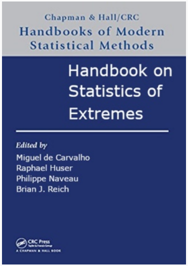
Reetam Majumder, University of Arkansas
ft. Neil Kpamegan, UMBC

University of Maryland, Baltimore County
Oct 31, 2025

Outline

- Part I: Basics of deep learning. Background, foundations, rules of thumb, common applications.
- Part II: Using deep learning for statistical analysis. Time series analysis, quantile regression, density estimation, amortized inference for intractable models
- Part III: Further topics. Ongoing research avenues, curios, and resources.

Outline



- Much of the content in parts I and II is based on an AI for extremes short course that Jordan Richards (Edinburgh), Likun Zhang (Mizzou) and I ran during the Contemporary Advances in Statistics of Extremes workshop held at Mizzou in June 2025.
- Deep learning has seen a lot of adoption in the extremes community (keep an eye out for the upcoming handbook!)
- Vignette at `https://github.com/reetamm/AI4stats`.

Preliminaries: Deep learning packages

- The two most popular deep learning packages are `tensorflow` (Abadi et al., 2015) and `torch` (Paszke et al., 2017)
- `tensorflow` was written in Python, while `torch` was written in `C++` with the python version known as `pytorch`
- For the most part, their development happens on `Python`, and hundreds (if not more?) packages use them as their basis
- `tensorflow` has a higher level API called `keras`, which simplifies a lot of the technical aspects.
- `torch` has something similar, called `luz`.
- Both `tensorflow/keras` and `torch/luz` are available on `R`, but their implementations couldn't be more different.
- `tf` on `R` needs `Python`, while `torch` doesn't.

keras vs. torch on R

	`keras`	`torch`
Ease of installation	Complicated	Straightforward
Features	Several Python packages available through reticulate or conda	Limited to packages developed on R
Accessibility	keras makes things easy	Slowly getting better
GPU support	Yes	Yes
Overhead	High on personal systems, but easy to run on Google Colab	Harder to run on Colab, but easy on personal devices

- `tf/keras` on `R` is just a wrapper of the `Python` library made possible by `reticulate`
- Since Google Colab already has Python, it bypasses much of the difficulty and allows the use of `keras3` with relative ease.
- However, it does not support the older `keras` package any more.

Part I: Basics of deep learning

Regression: General Setting

Given:

- Response variable Y (often in $\mathbb{R}$, but could also be discrete classes);
- Covariates $X \in \mathbb{R}^q$.

We are interested in learning some **estimand** that describes the conditional distribution of $Y \mid \mathbf{X} = \mathbf{x}$.

Typical estimands of interest are:

- Class probabilities: $\Pr\{Y = k | \mathbf{X} = \mathbf{x}\}$ for $k \in \mathbb{N}$;
- Expectation: $\mathbb{E}[Y | \mathbf{X} = \mathbf{x}]$;
- Quantiles: $Q_x(\tau) = \inf\{y : F_{Y|\mathbf{X}}(y|x) \geq \tau\}$ for $\tau \in (0, 1)$;
- "Parameters" of $F_{Y|\mathbf{X}}$. Can include parameters in the standard finite-dimensional sense, but may also be interested in semi-/non-parametric model for $F_{Y|\mathbf{X}}$.

Regression: How do we estimate these parameters?

Set up an estimable model $\mathbf{m} : \mathbb{R}^q \mapsto \boldsymbol{\Theta}$:

- Let $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ contain your estimands of interest.
- Setup a function $\mathbf{m}$ that maps your covariates $\mathbf{x}$ to $\boldsymbol{\theta}$, i.e., $\boldsymbol{\theta}(\mathbf{x}) := \mathbf{m}(\mathbf{x})$.
- Get yourself some data $\{(y_i, \mathbf{x}_i)\}_{i=1}^n$.

Estimate $\mathbf{m}$ via minimisation of some loss or cost function:

- Class probabilities: binary cross-entropy;
- Expectation: MSE $n^{-1} \sum_{i=1}^n (y_i - \theta(\mathbf{x}_i))^2$
- Quantiles: Pinball loss $n^{-1} \sum_{i=1}^n (y_i - \theta(\mathbf{x}_i))(\tau - \mathbb{1}\{y_i < \theta(\mathbf{x}_i)\})$;
- $F_{Y|\mathbf{X}}$. Associated negative log-likelihood.

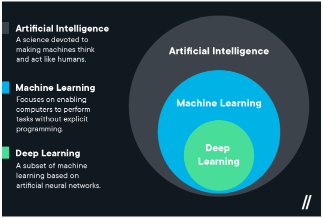
For example, an estimate for the conditional expectation function, say $\widehat{\theta(\mathbf{x})}$, is the minimiser

$$\widehat{\theta(\mathbf{x})} \in \arg\min_{\mathbf{m} \in \mathcal{M}} \frac{1}{n} \sum_{i=1}^{n} (y_i - \mathbf{m}(\mathbf{x}_i))^2,$$

where $\mathcal{M}$ is some appropriate space of estimable functions.

But what should $\mathcal{M}$ look like?

Machine learning

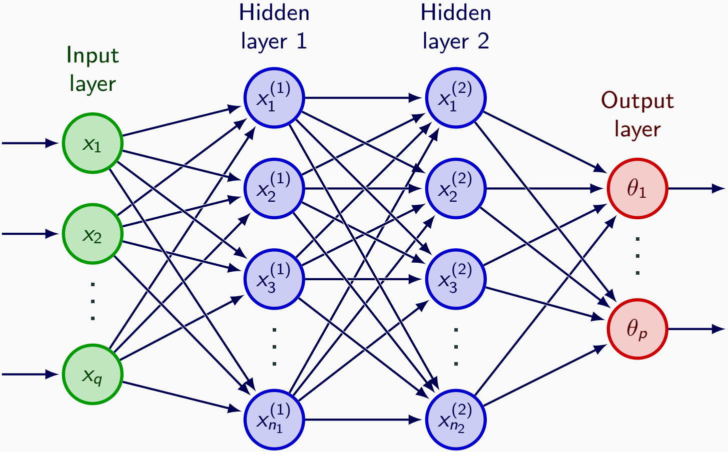


- The **fundamental** difference between statistics and AI/ML/DL is the complexity of $\mathbf{m}(\cdot)$.
- Traditional statistics want $\mathbf{m}(\cdot)$ simple, e.g., linear, parametric, splines, so that estimates are interpretable.
- ML/DL wants flexibility in $\mathbf{m}(\cdot)$.
- Other key ideas - optimisation, regularisation, testing - tend to be the same.

The **deep** in Deep learning comes from the depth of **m**:

- We represent **m** as a neural network. This is just a composition of **layers** of simple, differentiable operations.
- There are lots of different choices of operations, which give rise to different types of neural network. Today, we focus on the standard feed-forward multi-layered perceptron

Deep learning



The neural network $\mathbf{m}_{\psi}$ is constructed as a composition of $L \in \mathbb{N}$ *hidden layers*, $\mathbf{m}^{(l)}$ for $l = 1, \ldots, L$, and an *output/final* layer, $\mathbf{m}^{(L+1)}$, such that $\mathbf{m}_{\psi}(\cdot) := \mathbf{m}^{(L+1)} \circ \cdots \circ \mathbf{m}^{(1)}(\cdot)$.

- ψ contains all the estimable parameters (weights and biases);
- the neural network has the hierarchical structure

$$
\begin{aligned}
\mathbf{x}^{(1)} :=& \mathbf{m}^{(1)}(\mathbf{x}) = \mathbf{a}^{(1)}\left(\mathbf{W}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}\right), \\
\mathbf{x}^{(2)} :=& \mathbf{m}^{(2)}(\mathbf{x}^{(1)}) = \mathbf{a}^{(2)}\left(\mathbf{W}^{(2)}\mathbf{x}^{(1)} + \mathbf{b}^{(2)}\right), \\
&\vdots \\
\mathbf{x}^{(L)} :=& \mathbf{m}^{(L)}(\mathbf{x}^{(L-1)}) = \mathbf{a}^{(L)}\left(\mathbf{W}^{(L)}\mathbf{x}^{(L-1)} + \mathbf{b}^{(L)}\right), \\
\boldsymbol{\theta}(\mathbf{x}) :=& \mathbf{m}^{(L+1)}(\mathbf{x}^{(L)}) = \mathbf{a}^{(L+1)}\left(\mathbf{W}^{(L+1)}\mathbf{x}^{(L)} + \mathbf{b}^{(L+1)}\right).
\end{aligned}
$$

Activation functions

Activation functions provide flexibility by making the layers non-linear.

Popular choices includes:

- ReLU: $\mathbf{a}(\mathbf{x})_j = \max(x_j, 0)$
- sigmoid: $\mathbf{a}(\mathbf{x})_j = \exp(x_j)/(1 + \exp(x_j))$
- tanh
- eLU and leaky ReLU
- softmax: $\mathbf{a}(\mathbf{x}) = (\frac{\exp(x_1)}{\sum_i \exp(x_i)}, \frac{\exp(x_2)}{\sum_i \exp(x_i)}, \dots)$

Some common types of layers

- Dense layer
- Convolutional layer
- Maxpooling layer
-

Training models

Models are trained by solving the empirical risk minimisation problem

$$\widehat{\boldsymbol{\psi}} \in \arg\min_{\boldsymbol{\psi}} \frac{1}{n} \sum_{i=1}^{n} \ell\{y_i; \mathbf{m}_{\boldsymbol{\psi}}, (\mathbf{x}_i)\},$$

where ℓ is our loss function.

In **statistics**, this would be achieved using gradient descent, via the update

$$\boldsymbol{\psi} \leftarrow \boldsymbol{\psi} - \frac{\lambda}{n} \sum_{i=1}^{n} \boldsymbol{\nabla}_{\boldsymbol{\psi}} \ell\{y_i, \mathbf{m}_{\boldsymbol{\psi}}(\mathbf{x}_i)\}, \tag{1}$$

where $\lambda > 0$ is a tunable learning rate and $\boldsymbol{\nabla}_{\boldsymbol{\psi}}$ denotes the differential operator with respect to entries in $\boldsymbol{\psi}$.

Training deep models

In **deep learning**, where n is typically large, a more economical approach is to update over individual observations (SGD), using the rule

$$\boldsymbol{\psi} \leftarrow \boldsymbol{\psi} - \lambda \boldsymbol{\nabla}_{\boldsymbol{\psi}} \ell\{y_i, \mathbf{g}_{\boldsymbol{\psi}}(\mathbf{x}_i)\}, \qquad i = 1, \ldots, n. \tag{2}$$

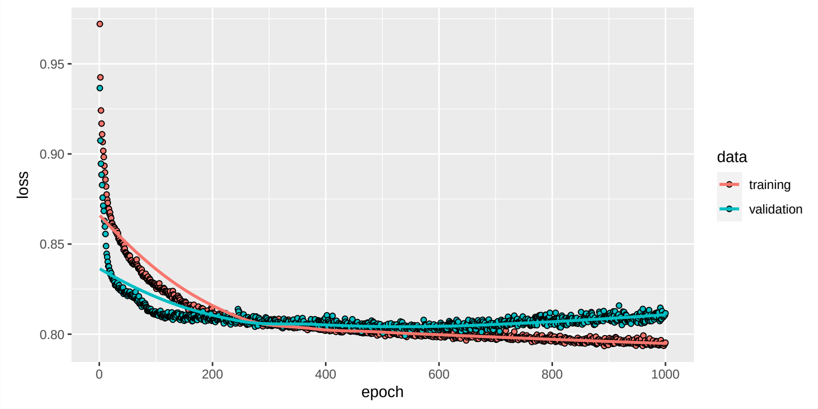
Neural networks are usually trained using an algorithm that falls in-between gradient descent and true SGD, which is called *mini-batch SGD*. In this case, all samples are split into n_b mini-batches of equal size, and n_b updates are performed for every **epoch** of the training scheme.

Usually, $\boldsymbol{\nabla}_{\boldsymbol{\psi}}$ is tractable and fast-to-compute via **backpropagation**. This is because the neural network model is a composition of simple differentiable operations.

Regularisation: Training/validation/testing

Due to their large number of parameters, neural networks are prone to overfitting. It's necessary to use **validation/testing** to assess overfitting.

- Subset the data into training, validation, and testing.
- Each set has a different purpose:
    - The model is **trained** on the training data. This is used to update the weights and biases.
    - We **validate** the model on the validation data. Typically used for model selection.
    - To get a truly unbiased evaluation of the model fit and to compare amongst different models, we **test** the model on test data.



loss
0.95
0.90
0.85
0.80
0
200
400
600
800
1000
epoch
data
training
validation

Regularisation/numerics

Other forms of regularisation to consider:

- Weight penalties (think LASSO or ridge)
- Random weight initialization
- Dropout
- Sharing weights*
- Early-stopping* and checkpoints*

Some numerical aspects:

- Pre-training*
- Standardising input data*
- Parameter dependent support
- Optimizing architecture/training scheme (please don't ask about this)

Part II: Deep Learning for Statistical Analysis

Deep GPD regression

We now use these ideas to perform deep GPD regression. We perform a **two-stage** estimation procedure:

- Use the tilted loss to estimate $u(\mathbf{x})$ as the τ-quantile of $Y \mid \mathbf{x} = \mathbf{x}$ (for τ close to one).
- Model conditional excesses $(Y - u(\mathbf{x})) \mid (\mathbf{X} = \mathbf{x})$ as $\text{GPD}(\sigma_u(\mathbf{x}), \xi(\mathbf{x}))$.

The latter stage follows via maximum likelihood and we consider three different models for $(\sigma_u(\mathbf{x}), \xi(\mathbf{x}))$.

Semi-parametric quantile regression (SPQR)

Estimating entire densities

- Our goal almost always is to make probability statements based on a fitted model
- However, sometimes we *don't* want to assume a parametric form of the density
- In that case, we need:
  - A non/semi-parametric way to represent the density
  - A loss function which evaluates the (usual negative log-likelihood) loss between the response y_i and the estimates $\pi(x_i)$
- SPQR does this using an $M-$spline representation of continuous density functions

SPQR

SPQR represents a continuous distribution $Y|X$ as:

$$f(y|x) = \sum_{k=1}^{K} \pi_k(x) M_k(y)$$

- $\{\pi_k\}_{k=1}^{K}$ is a vector of probabilities output from a NN
- The loss function is $-\log f(y|x)$
- $M-$splines are rescaled $B-$splines
- $I-$splines are just integrated $M-$splines such that:

$$F(y|x) = \sum_{k=1}^{K} \pi_k(x) I_k(y)$$

- Representation is scale-invariant

The good, the bad, and the next.

While the demo uses `keras`, there *is* a package based on `torch` at `https://github.com/stevengxu/SPQR` (Xu et al., 2022).
Good.

- Supports multi-modal input and various architectures
- Quantile function can be estimated using interpolation
- Can easily make probability statements

Bad.

- $M-$splines are bounded - extrapolation outside the observed range of y not possible

Next up.

- SPQRx (Majumder and Richards, 2025)

Part III: Further topics

Further topics

Machine learning for extreme quantile/extremal regression:

- NN-based methods:
  - GEV: (Cannon, 2010; Vasiliades et al., 2015; Bennett et al., 2015; Richards and Huser, 2024)
  - GPD: (Carreau and Bengio, 2007; Carreau et al., 2009; Carreau and Vrac, 2011; Richards et al., 2023; Allouche et al., 2024; Dahal et al., 2024; Pasche and Engelke, 2024; Majumder and Richards, 2025; Wilson et al., 2022)
  - Other: (Richards and Huser, 2022; Cisneros et al., 2024)
- Classical ML:
  - Random forests: (Taillardat et al., 2019; Gnecco et al., 2024)
  - Regression trees: (Farkas et al., 2021, 2024)
  - Boosting: (Velthoen et al., 2023; Koh, 2023; Koh et al., 2025)

Further topics

Multivariate and spatial models:

- Geometric methods: (Murphy-Barltrop et al., 2024; Mackay et al., 2025; De Monte et al., 2025)
- *d*-max-decreasing NNs: (Hasan et al., 2022)
- Deep compositional spatial models: (Shao et al., 2025)
- Generative methods:
  - Excess-GAN :(Allouche et al., 2025)
  - Ex-GAN: (Bhatia et al., 2021)
  - EV-GAN: (Allouche et al., 2022)
  - GPDflow: (Hu and Castro-Camilo, 2025)
  - EVT-GAN: (Boulaguiem et al., 2022)
  - HT-GAN: (Girard et al., 2024)
  - GP-GAN (Li et al., 2024)
  - Angular simulation: (Wessel et al., 2025)
  - WA-GAN: (Lhaut et al., 2025)
  - VAEs: (Lafon et al., 2023; Zhang et al., 2023)

Further topics

Statistical (unsupervised) learning:

- Clustering: (Janßen and Wan, 2020; Bernard et al., 2013; Saunders et al., 2021; Bador et al., 2015; Rohrbeck and Tawn, 2021; Vignotto et al., 2021; Boulin et al., 2023)
- Dimension reduction: (Chautru, 2015; Drees and Sabourin, 2021; Cooley and Thibaud, 2019)
- Anomaly detection: (Clifton et al., 2014; Rudd et al., 2017; Vignotto and Engelke, 2020)
- Review by Clémençon and Sabourin (2025)

Further topics

Fast inference:

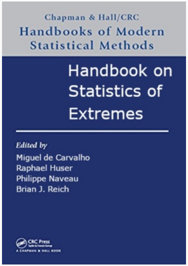
- Neural estimation:
  - Spatio-temporal: (Lenzi et al., 2023; Sainsbury-Dale et al., 2024a,b; Richards et al., 2024; Sainsbury-Dale et al., 2025; Dell'Oro and Gaetan, 2024; Rai et al., 2025; Hector and Lenzi, 2024)
  - Multivariate: (André et al., 2025; Hua, 2025)
  - Univariate: (Rai et al., 2024; Richards et al., 2025)
- Intractable likelihood approximation: (Walchessen et al., 2024; Majumder and Reich, 2023; Majumder et al., 2024)
- Variational inference: (Maceda et al., 2024)
- Dependence classification: (Ahmed et al., 2022; Wixson and Cooley, 2024)

Further topics

Other things to look out for:

- KANs: (de Carvalho et al., 2025)
- Conformal prediction: (Pasche et al., 2025)

Handbook chapters



- Ch11 - Principal Component Analysis for Multivariate Extremes. Cooley, Sabourin, and Wixson.
- Ch12 - Clustering Methods for Multivariate Extremes. Wan and Janßen.
- Ch20 - On the Simulation of Extreme Events with Neural Networks. Allouche, Girard, and Gobet.
- Ch21 - Extreme Quantile Regression with Deep Learning. Richards and Huser.
- Ch26 - Statistics of Extremes for Wildfires. Koh.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Ahmed, M., Maume-Deschamps, V., and Ribereau, P. (2022). Recognizing a spatial extreme dependence structure: A deep learning approach. *Environmetrics*, 33(4):e2714.

Allouche, M., Girard, S., and Gobet, E. (2022). Ev-gan: Simulation of extreme events with relu neural networks. *Journal of Machine Learning Research*, 23(150):1–39.

Allouche, M., Girard, S., and Gobet, E. (2024). Estimation of extreme quantiles from heavy-tailed distributions with neural networks. *Statistics and Computing*, 34(1):12.

Allouche, M., Girard, S., and Gobet, E. (2025). Excessgan: simulation above extreme thresholds using generative adversarial networks.

André, L. M., Wadsworth, J. L., and Huser, R. (2025). Neural bayes inference for complex bivariate extremal dependence models. *arXiv preprint arXiv:2503.23156*.

Bador, M., Naveau, P., Gilleland, E., Castellà, M., and Arivelo, T. (2015). Spatial clustering of summer temperature maxima from the CNRM-CM5 climate model ensembles & E-OBS over Europe. *Weather and Climate Extremes*, 9:17–24.

Bennett, K. E., Cannon, A. J., and Hinzman, L. (2015). Historical trends and extremes in boreal Alaska river basins. *Journal of hydrology*, 527:590–607.

Bernard, E., Naveau, P., Vrac, M., and Mestre, O. (2013). Clustering of maxima: Spatial dependencies among heavy rainfall in France. *Journal of Climate*, 26(20):7929–7937.

Bhatia, S., Jain, A., and Hooi, B. (2021). ExGAN: Adversarial generation of extreme samples. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 6750–6758.

Boulaguiem, Y., Zscheischler, J., Vignotto, E., van der Wiel, K., and Engelke, S. (2022). Modeling and simulating spatial extremes by combining extreme value theory with generative adversarial networks. *Environmental Data Science*, 1:e5.

Boulin, A., Di Bernardino, E., Laloë, T., and Toulemonde, G. (2023). Identifying regions of concomitant compound precipitation and wind speed extremes over Europe. *preprint arXiv:2311.11292*.

Cannon, A. J. (2010). A flexible nonlinear modelling framework for nonstationary generalized extreme value analysis in hydroclimatology. *Hydrological Processes: An International Journal*, 24(6):673–685.

Carreau, J. and Bengio, Y. (2007). A hybrid Pareto model for conditional density estimation of asymmetric fat-tail data. In *Artificial Intelligence and Statistics*, pages 51–58. PMLR.

Carreau, J., Naveau, P., and Sauquet, E. (2009). A statistical rainfall-runoff mixture model with heavy-tailed components. *Water resources research*, 45(10).

Carreau, J. and Vrac, M. (2011). Stochastic downscaling of precipitation with neural network conditional mixture models. *Water Resources Research*, 47(10).

Chautru, E. (2015). Dimension reduction in multivariate extreme value analysis. *Electronic Journal of Statistics*, 9(1):383–418.

Cisneros, D., Richards, J., Dahal, A., Lombardo, L., and Huser, R. (2024). Deep graphical regression for jointly moderate and extreme Australian wildfires. *Spatial Statistics*, 59:100811.

Clémençon, S. and Sabourin, A. (2025). Weak signals and heavy tails: Machine-learning meets extreme value theory. *arXiv preprint arXiv:2504.06984*.

Clifton, D. A., Clifton, L., Hugueny, S., and Tarassenko, L. (2014). Extending the generalised Pareto distribution for novelty detection in high-dimensional spaces. *Journal of Signal Processing Systems*, 74(3):323–339.

Cooley, D. and Thibaud, E. (2019). Decompositions of dependence for high-dimensional extremes. *Biometrika*, 106(3):587–604.

Dahal, A., Huser, R., and Lombardo, L. (2024). At the junction between deep learning and statistics of extremes: formalizing the landslide hazard definition. *Journal of Geophysical Research: Machine Learning and Computation*, 1(3):e2024JH000164.

de Carvalho, M., Ferrer, C., and Vallejos, R. (2025). A kolmogorov-arnold neural model for cascading extremes. *arXiv preprint arXiv:2505.13370*.

De Monte, L., Huser, R., Papastathopoulos, I., and Richards, J. (2025). Generative modelling of multivariate geometric extremes using normalising flows. *arXiv preprint arXiv:2505.02957*.

Dell'Oro, L. and Gaetan, C. (2024). Flexible space-time models for extreme data. *arXiv preprint arXiv:2411.19184*.

Drees, H. and Sabourin, A. (2021). Principal component analysis for multivariate extremes. *Electronic Journal of Statistics*, 15(1):908–943.

Farkas, S., Heranval, A., Lopez, O., and Thomas, M. (2024). Generalized Pareto regression trees for extreme event analysis. *Extremes*, pages 1–41.

Farkas, S., Lopez, O., and Thomas, M. (2021). Cyber claim analysis using generalized pareto regression trees with applications to insurance. *Insurance: Mathematics and Economics*, 98:92–105.

Girard, S., Gobet, E., and Pachebat, J. (2024). Deep generative modeling of multivariate dependent extremes.

Gnecco, N., Terefe, E. M., and Engelke, S. (2024). Extremal random forests. *Journal of the American Statistical Association*, 119(548):3059–3072.

Hasan, A., Elkhalil, K., Ng, Y., Pereira, J. M., Farsiu, S., Blanchet, J., and Tarokh, V. (2022). Modeling extremes with d-max-decreasing neural networks. In *Uncertainty in Artificial Intelligence*, pages 759–768. PMLR.

Hector, E. C. and Lenzi, A. (2024). When the whole is greater than the sum of its parts: Scaling black-box inference to large data settings through divide-and-conquer. *arXiv preprint arXiv:2412.20323*.

Hu, C. and Castro-Camilo, D. (2025). Gpdflow: Generative multivariate threshold exceedance modeling via normalizing flows. *arXiv preprint arXiv:2503.11822*.

Hua, L. (2025). Amortized neural inference on bivariate tail dependence and tail asymmetry. *Available at SSRN 5287687*.

Janßen, A. and Wan, P. (2020). k-means clustering of extremes. *Electronic Journal of Statistics*, 14(1):1211 – 1233.

Koh, J. (2023). Gradient boosting with extreme-value theory for wildfire prediction. *Extremes*, 26(2):273–299.

Koh, J., Steinfeld, D., and Martius, O. (2025). Using spatial extreme-value theory with machine learning to model and understand spatially compounding weather extremes. *Proceedings of the Royal Society A*, 481(2316):20240763.

Lafon, N., Naveau, P., and Fablet, R. (2023). A vae approach to sample multivariate extremes. *arXiv preprint arXiv:2306.10987*.

Lenzi, A., Bessac, J., Rudi, J., and Stein, M. L. (2023). Neural networks for parameter estimation in intractable models. *Computational Statistics & Data Analysis*, 185:107762.

Lhaut, S., Rootzén, H., and Segers, J. (2025). Wasserstein-aitchison gan for angular measures of multivariate extremes. *arXiv preprint arXiv:2504.21438*.

Li, J., Li, D., Li, P., and Samorodnitsky, G. (2024). Generalized pareto gan: Generating extremes of distributions. In *2024 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE.

Maceda, E., Hector, E. C., Lenzi, A., and Reich, B. J. (2024). A variational neural bayes framework for inference on intractable posterior distributions. *arXiv preprint arXiv:2404.10899*.

Mackay, E., Murphy-Barltrop, C. J., Richards, J., and Jonathan, P. (2025). Deep learning of joint extremes of metocean variables using the SPAR model. *preprint arXiv:2412.15808*.

Majumder, R. and Reich, B. J. (2023). A deep learning synthetic likelihood approximation of a non-stationary spatial model for extreme streamflow forecasting. *Spatial Statistics*, 55:100755.

Majumder, R., Reich, B. J., and Shaby, B. A. (2024). Modeling extremal streamflow using deep learning approximations and a flexible spatial process. *The Annals of Applied Statistics*, 18(2):1519–1542.

Majumder, R. and Richards, J. (2025). Semi-parametric bulk and tail regression using spline-based neural networks. *arXiv preprint arXiv:2504.19994*.

Murphy-Barltrop, C. J., Majumder, R., and Richards, J. (2024). Deep learning of multivariate extremes via a geometric representation. *preprint arXiv:2406.19936*.

Pasche, O. C. and Engelke, S. (2024). Neural networks for extreme quantile regression with an application to forecasting of flood risk. *The Annals of Applied Statistics*, 18(4):2818–2839.

Pasche, O. C., Lam, H., and Engelke, S. (2025). Extreme conformal prediction: Reliable intervals for high-impact events. *arXiv preprint arXiv:2505.08578*.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

References vii

Rai, S., Hoffman, A., Lahiri, S., Nychka, D. W., Sain, S. R., and Bandyopadhyay, S. (2024). Fast parameter estimation of generalized extreme value distribution using neural networks. *Environmetrics*, 35(3):e2845.

Rai, S., Nychka, D. W., and Bandyopadhyay, S. (2025). Modeling spatial extremes using non-gaussian spatial autoregressive models via convolutional neural networks. *arXiv preprint arXiv:2505.03034*.

Richards, J., Alotaibi, N., Cisneros, D., Gong, Y., Guerrero, M. B., Redondo, P. V., and Shao, X. (2025). Modern extreme value statistics for utopian extremes. eva (2023) conference data challenge: Team yalla. *Extremes*, 28(1):149–171.

Richards, J. and Huser, R. (2022). Regression modelling of spatiotemporal extreme US wildfires via partially-interpretable neural networks. *arXiv:2208.07581*.

Richards, J. and Huser, R. (2024). Extreme quantile regression with deep learning. In de Carvalho, M., Huser, R., Naveau, P., and Reich, B. J., editors, *Handbook on Statistics of Extremes*. Chapman & Hall / CRC.

Richards, J., Huser, R., Bevacqua, E., and Zscheischler, J. (2023). Insights into the drivers and spatiotemporal trends of extreme Mediterranean wildfires with statistical deep learning. *Artificial Intelligence for the Earth Systems*, 2(4):e220095.

Richards, J., Sainsbury-Dale, M., Zammit-Mangion, A., and Huser, R. (2024). Likelihood-free neural Bayes estimators for censored inference with peaks-over-threshold models. *Journal of Machine Learning Research*, 25(390):1–49.

Rohrbeck, C. and Tawn, J. A. (2021). Bayesian spatial clustering of extremal behavior for hydrological variables. *Journal of Computational and Graphical Statistics*, 30(1):91–105.

Rudd, E. M., Jain, L. P., Scheirer, W. J., and Boult, T. E. (2017). The extreme value machine. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 40(3):762–768.

Sainsbury-Dale, M., Zammit-Mangion, A., Cressie, N., and Huser, R. (2025). Neural parameter estimation with incomplete data. *arXiv preprint arXiv:2501.04330*.

Sainsbury-Dale, M., Zammit-Mangion, A., and Huser, R. (2024a). Likelihood-free parameter estimation with neural Bayes estimators. *The American Statistician*, 78(1):1–14.

Sainsbury-Dale, M., Zammit-Mangion, A., Richards, J., and Huser, R. (2024b). Neural bayes estimators for irregular spatial data using graph neural networks. *Journal of Computational and Graphical Statistics*, (just-accepted):1–28.

Saunders, K., Stephenson, A., and Karoly, D. (2021). A regionalisation approach for rainfall based on extremal dependence. *Extremes*, 24:215–240.

Shao, X., Richards, J., and Huser, R. (2025). Modeling nonstationary extremal dependence via deep spatial deformations. *arXiv preprint arXiv:2505.12548*.

Taillardat, M., Fougères, A.-L., Naveau, P., and Mestre, O. (2019). Forest-based and semiparametric methods for the postprocessing of rainfall ensemble forecasting. *Weather and Forecasting*, 34(3):617–634.

Vasiliades, L., Galiatsatou, P., and Loukas, A. (2015). Nonstationary frequency analysis of annual maximum rainfall using climate covariates. *Water Resources Management*, 29(2):339–358.

Velthoen, J., Dombry, C., Cai, J.-J., and Engelke, S. (2023). Gradient boosting for extreme quantile regression. *Extremes*, 26(4):639–667.

Vignotto, E. and Engelke, S. (2020). Extreme value theory for anomaly detection–the GPD classifier. *Extremes*, 23(4):501–520.

Vignotto, E., Engelke, S., and Zscheischler, J. (2021). Clustering bivariate dependencies of compound precipitation and wind extremes over Great Britain and Ireland. *Weather and Climate Extremes*, 32:100318.

Walchessen, J., Lenzi, A., and Kuusela, M. (2024). Neural likelihood surfaces for spatial processes with computationally intensive or intractable likelihoods. *Spatial Statistics*, 62:100848.

Wessel, J. B., Murphy-Barltrop, C. J., and Simpson, E. S. (2025). A comparison of generative deep learning methods for multivariate angular simulation. *arXiv preprint arXiv:2504.21505*.

Wilson, T., Tan, P.-N., and Luo, L. (2022). DeepGPD: A deep learning approach for modeling geospatio-temporal extreme events. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4245–4253.

Wixson, T. P. and Cooley, D. (2024). Neural network for asymptotic dependence/ independence classification: A series of experiments. *preprint doi.org/10.21203/rs.3.rs-3994810/v1*.

Xu, S. G., Majumder, R., and Reich, B. J. (2022). SPQR: An R package for semi-parametric density and quantile regression. https://arxiv.org/abs/2210.14482.

Zhang, L., Ma, X., Wikle, C. K., and Huser, R. (2023). Flexible and efficient spatial extremes emulation via variational autoencoders. *preprint arXiv:2307.08079*.