Beyond Point Prediction: Capturing Zero-Inflated & Heavy-Tailed Spatiotemporal Data with Deep Extreme Mixture Models

Tyler Wilson
wils1270.msu@gmail.com
Michigan State University
East Lansing, Michigan, USA

Andrew McDonald
mcdon499@msu.edu
Michigan State University
East Lansing, Michigan, USA

Asadullah Hill Galib
galibasa@msu.edu
Michigan State University
East Lansing, Michigan, USA

Pang-Ning Tan
ptan@msu.edu
Michigan State University
East Lansing, Michigan, USA

Lifeng Luo
lluo@msu.edu
Michigan State University
East Lansing, Michigan, USA

ABSTRACT

Zero-inflated, heavy-tailed spatiotemporal data is common across science and engineering, from climate science to meteorology and seismology. A central modeling objective in such settings is to forecast the intensity, frequency, and timing of extreme and non-extreme events—yet in the context of deep learning, this objective presents several key challenges. First, a deep learning framework applied to such data must unify a mixture of distributions characterizing the zero events, moderate events, and extreme events. Second, the framework must be capable of enforcing parameter constraints across each component of the mixture distribution. Finally, the framework must be flexible enough to accommodate for any changes in the threshold used to define an extreme event after training. To address these challenges, we propose Deep Extreme Mixture Model (*DEMM*), fusing a deep learning-based hurdle model with extreme value theory to enable point and distribution prediction of zero-inflated, heavy-tailed spatiotemporal variables. The framework enables users to dynamically set a threshold for defining extreme events at inference-time without the need for retraining. We present an extensive experimental analysis applying *DEMM* to precipitation forecasting, and observe significant improvements in point and distribution prediction All code is available at https://github.com/andrewmcdonald27/DeepExtremeMixtureModel.

CCS CONCEPTS

• **Computing methodologies** → **Neural networks**; *Regularization*; • **Mathematics of computing** → *Maximum likelihood estimation*; • **Applied computing** → **Earth and atmospheric sciences**.

KEYWORDS

Zero-Inflated, Heavy-Tailed, Spatiotemporal Modeling, Extreme Value Theory, Deep Mixture Model

KDD '22, August 14–18, 2022, Washington, DC, USA

ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539464

ACM Reference Format:
Tyler Wilson, Andrew McDonald, Asadullah Hill Galib, Pang-Ning Tan, and Lifeng Luo. 2022. Beyond Point Prediction: Capturing Zero-Inflated & Heavy-Tailed Spatiotemporal Data with Deep Extreme Mixture Models . In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3534678.3539464

```
$ENV{'TZ'}='America/New_York';
```

1 INTRODUCTION

Spatiotemporal variables of interest in science and engineering often exhibit two distinct characteristics: (i) zero-inflation, in which there is an abundance of values exactly equal to zero or within measurement error of zero; and (ii) heavy-tailedness, in which extreme values beyond a threshold of some physical significance arise. In meteorology, for example, zero-inflation and heavy-tailedness naturally partitions the global distribution of rainfall into a three-component structure—zero, moderate, and extreme events. Specifically, over time and space, many precipitation measurements will be exactly zero, some will be moderate (nonzero and nonextreme), and a significant few will be extreme. A similar structure arises in seismology, where a large proportion of readings at a given time and location are within noise tolerance of zero, with minor and moderate earthquakes constituting moderate values, and rare yet significant large earthquakes constituting extreme values.

As deep learning continues to be adopted in a variety of spatiotemporal applications [6, 19, 20], the importance of accurately modeling all three components of a non-normally-distributed spatiotemporal variable will only continue to increase. Accurate forecasts of extremes, for instance, could empower weather agencies to warn residents in advance of a flood and empower seismologists to warn residents of an impending large earthquake. On the other hand, accurate modeling of zero-valued and moderate-valued events is necessary for day-to-day operations and should not be sacrificed at the expense of extreme-valued events. A deep learning framework capable of balancing predictive performance across all three event classes is therefore essential. Yet developing such a deep learning framework to accurately predict the intensity, frequency, and timing of zero events, moderate events, and extreme events proves challenging for a number of reasons.

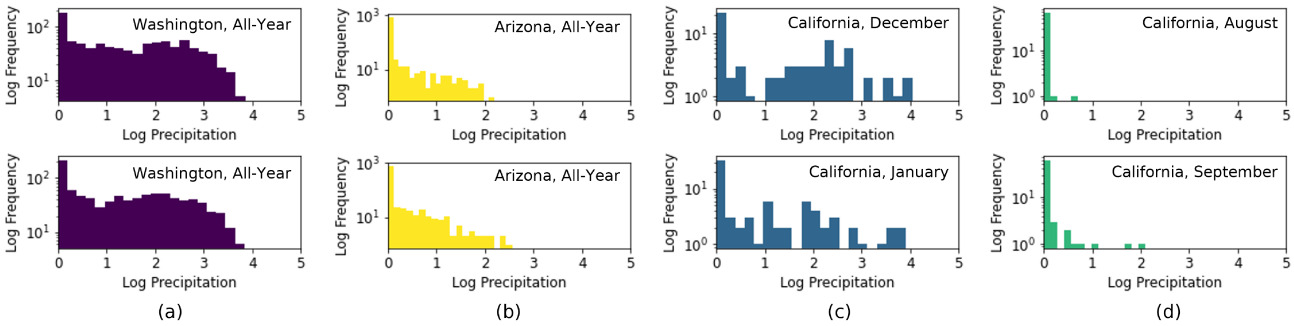

Figure 1: Spatial and temporal autocorrelation of precipitation challenges conventional deep learning frameworks. The first two columns depict similarity of precipitation distributions at neighboring grid cells (1° latitude apart) in (a) WA and (b) AZ; yet there is clear spatial heterogeneity between precipitation in WA and AZ. Similarly, the distributions of precipitation in CA for (c) Dec/Jan and (d) Aug/Sep are similar; yet there is clear temporal heterogeneity between winter and summer.

First, the probability distributions governing the zero, moderate, and extreme events have distinct characteristics, constraints, and parameters that must be estimated from data. Yet, these three distributions must be connected to generate predictions approximating a continuous, real-valued target variable, necessitating the construction of a well-defined mixture model from which conditional mean and confidence interval predictions may be derived. Unifying the three distributions within a single differentiable deep learning framework presents the first major challenge to be addressed.

Second, the probability distribution used to model extreme events exhibits natural constraints on its parameters. These constraints must be enforced to ensure a model's fidelity in characterizing the tail distribution of a random variable, leading to numerical instability without an appropriate reparameterization scheme. Designing an appropriate reparameterization scheme to enable constraint enforcement of the probability model inferred by a deep neural network presents the second major challenge to consider.

Finally, the model predictions may depend on the choice of threshold used to define extreme events. It is likely that users will want to vary this threshold when making predictions to test different scenarios. Yet in general, a model trained on a specific threshold for extreme values may not necessarily perform well when applied to different thresholds. Thus, developing a robust framework which enables users to dynamically set the extreme threshold at runtime without requiring the model to be retrained presents the third major challenge to be investigated in this study.

In addition to these major challenges, a model trained to fit the spatiotemporal data must account for spatial and temporal autocorrelation (Figure 1), non-linear interactions, feedback effects, and teleconnections. Properly leveraged, these features can improve model performance: for instance, spatial and temporal autocorrelation dictate that it is likely for extreme precipitation to be recorded at a location conditional on the knowledge that extreme precipitation was recorded nearby. Improperly ignored, these features can lead a predictive model astray. Accounting for the unique structure present in spatiotemporal data when developing our framework is thus a fourth challenge we consider.

To address these challenges, this paper presents a novel Deep Extreme Mixture Model (*DEMM*), fusing a deep learning-based hurdle model with extreme value theory (EVT) to predict point and distribution estimates of a spatiotemporal variable taking zero, nonzero-nonextreme, and extreme values. A typical hurdle model is a mixture model with two underlying components, one governing the strictly zero values and a second governing the distribution of its nonzero values. *DEMM* extends this paradigm by incorporating a third component using the generalized Pareto (GP) distribution to model the distribution of extreme values above a specified threshold. *DEMM* captures spatial and temporal relationships within the data using a 3D convolutional neural network (CNN), while an objective function consisting of root mean squared error (RMSE) and negative log-likelihood (NLL) terms is used in training to encourage sensible point and distribution predictions.

To summarize, the main contributions of this work are as follows.

(i) We propose *DEMM*, a novel framework capable of producing point and distribution estimates of the intensity, frequency, and timing of zero events, moderate events, and extreme events in non-normally-distributed spatiotemporal data.
(ii) We propose a novel reparameterization which ensures that the GP distribution used to model extremes is well-defined with valid parameters.
(iii) We propose a technique for allowing the user to dynamically alter their chosen extreme event threshold at test time without retraining the model.
(iv) We demonstrate the effectiveness of the *DEMM* in predicting the intensity, frequency, and timing of extreme events on a real world precipitation dataset.

2 RELATED WORKS

An early combination of neural networks (NNs) with mixture models, known as Mixture Density Networks, was developed to represent arbitrarily learned conditional distributions, from which point predictions could be derived [5]. More recently, deep neural networks (DNNs) have been integrated with mixture models for a wider range of purposes. Zong et al. [28] propose a Deep Autoencoding Gaussian Mixture Model (*DAGMM*) for unsupervised

anomaly detection, in which a DNN-based autoencoder maps data into a low-dimensional latent space and the joint density of the data in latent space is represented by a Gaussian mixture model. Viroli and McLachlan [25] propose a Deep Gaussian Mixture Model (*DGMM*) for clustering, in which variables at each layer of a DNN are encouraged to follow a mixture of Gaussian distributions.

Hurdle models, in which a nonnegative real-valued random variable is modeled as a mixture of a zero-valued component and a positive-valued component, were originally proposed by Cragg [9] for application to economic data. Closely-related is the zero-inflated Poisson model in which a nonnegative discrete-valued random variable is modeled as a zero-nonzero mixture, originally proposed by Lambert [16] and applied to manufacturing data. Continuous and discrete hurdle models have since found several applications in spatiotemporal settings, including precipitation forecasting [1, 2], species abundance estimation [3], and wildfire occurrence [21].

More recently, DNNs have been applied to estimate the parameters of hurdle models. Kong et al. [15] propose a deep hurdle network for multi-target regression and demonstrate its application to multiple species abundance estimation. Vandal et al. [24] use a deep neural network to estimate the parameters of a hurdle model using a log-normal distribution as its second component and leverage Monte-Carlo dropout to account for uncertainty in model parameters in an application to precipitation downscaling (super-resolution). Bacry et al. [13] propose a deep learning-based Zero-inflated Mixture of Multinomial distributions (*ZiMM*) model and apply it to modeling long-term post-surgery adverse events in a medical database.

Traditional statistical approaches have long been used to infer the distribution of extreme values [8]. However, these traditional approaches generally assume there is a relatively simple relationship between predictors and the parameters governing the generalized Pareto distribution used to model extreme values. In addition, these traditional approaches fail to model the full distribution of the data, focusing exclusively on the distribution of extremes instead. More recent approaches combining deep learning with EVT have also been limited in scope, failing to tightly integrate two paradigms. Instead, EVT is used as a post-processing step [26, 27] or utilized in a limited manner [4, 10]. Exceptions in which EVT is more tightly integrated into the fabric of a deep learning framework include [11, 17], but these works do not address the task of spatiotemporal distribution and point prediction unlike the work considered here.

3 PRELIMINARIES

This section formalizes the problem statement and provides a brief introduction to extreme value theory (EVT) and the hurdle model, both of which are integral to the proposed *DEMM* framework.

3.1 Problem Statement

Let $\mathcal{D} = \{(X_{lw}, Y_{lw}) \mid w \in \{1, \cdots, W\};\ l \in \{1, \cdots, L\}\}$ be a spatiotemporal dataset, where $X_{lw} \in \mathbb{R}^{d \times \tau}$ denotes the sequence of d predictors for a prediction window w of width τ at location l while $Y_{lw} \in \mathbb{R}$ denotes the value of the target variable associated with the prediction window. The locations are assumed to be organized onto a spatial grid $\mathcal{S}$ while each prediction window w is associated with a sequence of discrete time steps, $[t_1^w, t_2^w, \cdots, t_\tau^w]$. Thus, each $X_{lwi} \in \mathbb{R}^d$ represents a vector of predictors at a particular location l at time step $t_i^{(w)}$ while Y_{lw} denotes the observed value at the last time step, t_τ^w. For brevity, let $X_{:wi} \in \mathbb{R}^{L \times d}$ denote a gridded snapshot image of the predictors across all spatial locations at time step $t_i^{(w)} \in w$ while $X_{:w} = \{X_{:wi} \mid i \in \{1, \cdots, \tau\}\}$ denotes a sequence of such snapshots. Similarly, the gridded snapshot of the target variable at time step $t_\tau^{(w)}$ will be denoted as $Y_{:w}$. To determine whether the value of the target variable is an extreme value, let U_{lw} be the excess threshold for location l and prediction window w. Any observation in the window w that exceeds this threshold will be considered extreme. The set of threshold values for all locations in a given prediction window w is denoted as $U_{:w}$. In practice, users will likely assign a constant value for U_{lw} in all prediction windows and possibly all locations but for full generality, we allow it to be defined separately at each prediction window and location.

3.2 Extreme Value Theory

There are two widely-used statistical distributions for modeling extremes—(1) the generalized extreme value (GEV) distribution, which is used to model the distribution of block maxima, and (2) the generalized Pareto (GP) distribution, which is used to model the distribution of excesses above a given threshold u [8]. Since we are primarily interested in the distributions of excesses over thresholds, this work will focus only on the generalized Pareto distribution.

The density function of the GP distribution is given by

$$P(y) = \begin{cases} \frac{1}{\sigma}\left[1 + \frac{\xi y}{\sigma}\right]^{-\frac{1}{\xi}-1}, & \xi \neq 0 \\ \frac{1}{\sigma} e^{-\frac{y}{\sigma}} & \xi = 0 \end{cases} \tag{1}$$

Observe that the distribution is characterized by two parameters: shape ξ, and scale σ, assuming threshold $u = 0$. Note that y may be replaced with $y - u$ when $u \neq 0$. There are two key constraints that must be satisfied by the GP distribution parameters, namely, a positivity constraint on its scale parameter σ and a more complex constraint involving the shape, scale, and samples from the distribution, i.e.:

$$\sigma > 0 \quad \text{and} \quad \forall y :\ 1 + \frac{\xi y}{\sigma} > 0 \tag{2}$$

Note that this second constraint is always satisfied when modeling precipitation data if $\xi \geq 0$ since $y \geq 0$. Another important fact about the GP distribution is that its expected value is given by $E[Y] = \frac{\sigma}{1-\xi}$ when $\xi < 1$ but is undefined otherwise.

3.3 Hurdle Model

Zero-inflated data are commonly found in many applications. When modeling daily precipitation, for instance, most days have zero rainfall. In these cases, a hurdle model can be used to separately model the probability that the variable is zero or nonzero. A hurdle model is a mixture model consisting of two components:

$$P(Y = y) = \begin{cases} p & y = 0 \\ (1 - p) \cdot f_Y(y) & y > 0 \end{cases} \tag{3}$$

where Y is the random variable, p is its probability of being 0 and f_Y is the probability density function of Y when its value is nonzero. Any valid probability density function can be used as the second

component of the hurdle model (i.e. f) as long as its integral over y from 0 to infinity is 1 since:

$$\int_0^\infty P(y)dy = \int_0^\infty \Big(p \cdot I[y=0] + (1-p) \cdot f_Y(y) \cdot I[y \neq 0]\Big)dy \tag{4}$$

$$= p + (1-p)\int_0^\infty f_Y(y) \cdot I[y \neq 0]dy \tag{5}$$

$$= p + (1-p)\int_{>0}^\infty f_Y(y)dy \tag{6}$$

where $I[\cdot]$ denotes the indicator function. Furthermore, since Y is a continuous random variable, the density function f_Y is zero at any given value of y, including $y = 0$. Hence $\int_0^\infty P(y)dy = 1$ as long as $\int_0^\infty f_Y(y)dy = 1$.

4 DEEP EXTREME MIXTURE MODEL

The core of the Deep Extreme Mixture Model (*DEMM*) is a mixture model which governs the conditional distribution of the target variable, Y_{lw}. Figure 2 presents a schematic illustration of the *DEMM* architecture, which can be divided into three major components. The first component is a 3D convolutional neural network, which is responsible for modeling the spatiotemporal relationships within the predictors in addition to inferring the impact of the choice of threshold on the overall distribution. The second component is a constraint enforcement module, which is responsible for transforming the output of the neural network, A_{lw}, into a feasible set of mixture model parameters, θ_{lw}. The third component corresponds to the mixture model itself. We will introduce the mixture model at the heart of the *DEMM* first before describing the rest of the components in detail.

4.1 Mixture Model

The *DEMM* is centered around estimating the parameters of a mixture model. The mixture model is a combination of three probability distributions, each of which is responsible for a different range of values for the target variable. The three components of the mixture model have a combined total of six parameters to be learned, which are unique for each window w and location l.

Because the model is intended for use with zero inflated data, such as precipitation, it is based on a hurdle model, extended to account for the modeling of extreme values. The first component of the mixture model corresponds to a Bernoulli distribution to estimate the probability the target variable has the value of zero. Since the variable of interest is assumed to be non-negative, this component corresponds to the lower boundary of the distribution.

The second component governs the distribution of nonzero values below a certain threshold, U_{lw}. For precipitation prediction, a truncated log-normal distribution with parameters μ_{lw} and s_{lw} can be used, though the *DEMM* framework can accommodate other types of density functions. The density function of a non-truncated log-normal distribution with parameters μ_{lw} and s_{lw} is given by:

$$\hat{f}_1(Y_{lw}; \mu_{lw}, s_{lw}) = \frac{1}{Y_{lw}\sigma_{lw}\sqrt{2\pi}} \exp\left(-\frac{(\log Y_{lw} - \mu_{lw})^2}{2\sigma_{lw}^2}\right). \tag{7}$$

where the subscript 1 of the function $\hat{f}_1$ denotes the second component of the mixture model. Let $\hat{F}_1$ be the cumulative distribution function of $\hat{f}_1$. The truncated log-normal distribution function can be expressed as follows:

$$f_1(Y_{lw}) = \frac{\hat{f}_1(Y_{lw})}{\hat{F}_1(U_{lw}; \mu_{lw}, s_{lw})} \tag{8}$$

with the domain $0 < Y_{lw} < U_{lw}$.

Together, the first two components of the mixture model are similar to a conventional hurdle model. However, a third component is needed to ensure that the mixture model fits well to the empirical distribution, especially at the tail end of the distribution. As we are interested in modeling excess values over a threshold, U_{lw}, the generalized Pareto distribution is chosen as the third component of the mixture model. This ensures that the model is well specified for large values of Y_{lw} that exceed U_{lw}. Its density function, denoted as f_2, is given in Equation (1), with parameters ξ_{lw} and σ_{lw}.

To ensure that its integral over the domain of Y_{lw} is equal to 1, the last two components underlying the mixture model must be rescaled. The lognormal component is rescaled by a factor of $(1 - p_{lw}^{(0)}) \cdot p_{lw}^{(1)}$, where $p_{lw}^{(0)}$ represents the probability that $Y_{lw} = 0$ and $p_{lw}^{(1)}$ represents the probability it is nonzero and will not exceed the threshold. The GP component must be rescaled by a factor of $(1 - p_{lw}^{(0)}) \cdot (1 - p_{lw}^{(1)})$. Thus, the full distribution of the mixture model used in *DEMM* is:

$$P(Y_{lw} \mid X_{:w}; U_{:w}; \theta_{lw})$$
$$= \begin{cases} p_{lw}^{(0)} & Y_{lw} = 0 \\ (1 - p_{lw}^{(0)}) \cdot p_{lw}^{(1)} \cdot f_1(Y_{lw}; \mu_{lw}, s_{lw}) & 0 < Y_{lw} < U_{lw} \\ (1 - p_{lw}^{(0)}) \cdot (1 - p_{lw}^{(1)}) \cdot f_2(Y_{lw}; \xi_{lw}, \sigma_{lw}) & U_{lw} \leq Y_{lw} \end{cases} \tag{9}$$

Collectively, the parameters of the mixture model are denoted as the following six-dimensional vector:

$$\theta_{lw} = (p_{lw}^{(0)}, p_{lw}^{(1)}, \mu_{lw}, s_{lw}, \xi_{lw}, \sigma_{lw}) \tag{10}$$

The target variable is a sample from the conditional distribution defined by this mixture model. Given the mixture model parameters, it is easy to compute the negative log likelihood loss as follows:

$$\begin{aligned} \mathcal{L}_{\text{NLL}} = -\sum_{lw} \Big(& I[Y_{lw} = 0] \cdot \log(p_{lw}^{(0)}) \\ & + I[0 < Y_{lw} < U_{lw}] \cdot \Big[\log(1 - p_{lw}^{(0)}) \\ & + \log(p_{lw}^{(1)}) + \log(f_1(Y_{lw}; \mu_{lw}, s_{lw})) \Big] \\ & + I[U_{lw} < Y_{lw}] \cdot \Big[\log(1 - p_{lw}^{(0)}) \\ & + \log(1 - p_{lw}^{(1)}) + \log(f_2(Y_{lw}; \xi_{lw}, \sigma_{lw})) \Big] \Big) \end{aligned} \tag{11}$$

In addition the expected value of the mixture model can be easily computed as a weighted sum of the component means:

$$\begin{aligned} \hat{Y}_{lw} = & \; p_{lw}^{(0)} \cdot 0 \\ & + (1 - p_{lw}^{(0)}) \cdot p_{lw}^{(1)} \cdot \exp(\mu_{lw} + s_{lw}^2/2) \cdot \frac{\Phi\left[\frac{\ln(U_{lw}) - \mu_{lw} - s_{lw}^2}{s_{lw}}\right]}{\Phi\left[\frac{\ln(U_{lw}) - \mu_{lw}}{s_{lw}}\right]} \\ & + \; (1 - p_{lw}^{(0)}) \cdot (1 - p_{lw}^{(1)}) \cdot \left[U_{lw} + \frac{\sigma_{lw}}{1 - \xi_{lw}}\right] \end{aligned} \tag{12}$$

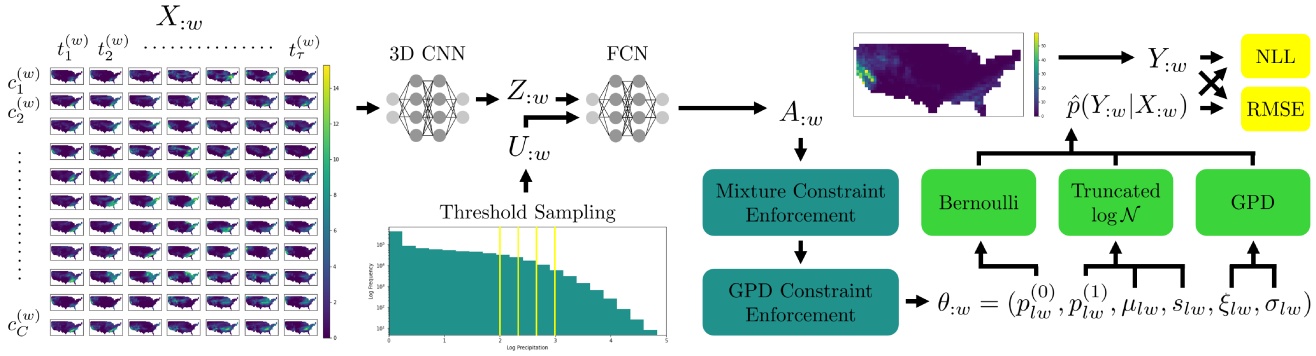

Figure 2: An overview of the proposed *DEMM* architecture. A 3D CNN capturing spatiotemporal relationships maps input $X_{:w}$ to features $Z_{:w}$, which are combined with randomly sampled thresholds $U_{:w}$ and passed to a FCN to compute activations $A_{:w}$. Constraint enforcement is then applied to extract $\theta_{:w}$ parameterizing a three-component mixture model, trained to minimize a weighted sum of negative log-likelihood and root mean squared error.

where Φ is the cumulative distribution function of a standard normal distribution. The value $\hat{Y}_{lw}$ can be used as a point prediction.

4.2 Deep Neural Network

This section introduces the deep neural network architecture that is responsible for modeling spatiotemporal relationships among the predictors to generate output activations that can be used to estimate the distribution parameters, θ_{lw}, given in Equation (10). As the distribution parameters depend on the threshold used to define extreme events, a key innovation of *DEMM* is its flexibility in allowing the choice of threshold to be varied at inference time without requiring the model to be retrained. To provide such flexibility, the DNN learns the distribution parameters as a function of input predictors $X_{:w}$ and thresholds $U_{:w}$. These thresholds can be varied randomly during training as will be described in Section 4.4.

The mapping from the input predictors and thresholds for a particular window to the final activation is performed in two stages. First, a 3D convolutional neural network (3D CNN) is used to learn a feature representation of the spatio-temporal input predictors. 3D CNNs [12] are generalizations of the 2D convolutions conventionally used in image processing and are applied here across latitude, longitude, and time. The 3D CNN works by computing the inner product between a filter of parameters with small localized regions within the 3D spatiotemporal volume. These convolutional layers alternate with non-linear activation functions such as the ReLU or tanh functions. Similar 3D CNNs have had success in other spatiotemporal applications [7, 12, 18, 22, 23]. The application of the 3D CNN to the predictors can be written formally as $z_{:w} = g(X_{:w})$ where $z_{:w} \in R^{L \times d}$ and g is the non-linear function associated with the trained 3D CNN.

Once the spatiotemporal features have been extracted, they are combined with the chosen threshold at each location so that the parameters of the mixture model can be estimated. This is accomplished by concatenating the threshold at each location, U_{lw}, to the location's spatiotemporal feature representation, z_{lw}, and feeding the results to a fully connected neural network (FCN). The output of the network is $A_{lw} = (A_{lw}^{(1)}, A_{lw}^{(2)}, \cdots A_{lw}^{(6)}) = h(z_{lw}, U_{lw}) \in \mathbb{R}^6$, where h represents the non-linear function associated with the fully connected neural network. Observe that each location and window are processed separately by the fully connected neural network.

4.3 Constraint Enforcement

Since the output of the deep neural network is completely unconstrained, it may not be suitable for use as parameters of the mixture model, which must satisfy certain feasibility conditions including the GP inequality constraints in Equation (2). Specifically, the mixture probabilities $p_{lw}^{(0)}$ and $p_{lw}^{(1)}$ are constrained to be between 0 and 1, s_{lw} and σ_{lw} are constrained to be non-negative, and ξ_{lw} and σ_{lw} must jointly satisfy the following inequality [8]:

$$\forall Y_{lw} : 1 + \frac{\xi_{lw} Y_{lw}}{\sigma_{lw}} > 0 \tag{13}$$

Given that the mean of the mixture model $\hat{Y}_{lw}$ will be used as a point estimate of Y_{lw}, this requires computing the mean of the three components of the mixture model. However, the mean of the GP distribution is only well-defined when $\xi_{lw} < 1$. This imposes another constraint that needs to be satisfied.

The constraint enforcement module transforms the output activation of the neural network, A_{lw}, into parameters of the mixture model, θ_{lw}, such that all the constraints are satisfied. First, $\mu_{lw} = A_{lw}^{(3)}$ is unconstrained. The constraints on $p_{lw}^{(0)}$ and $p_{lw}^{(1)}$ are easy to achieve by passing the corresponding activations through a sigmoid function, $\sigma[\cdot]$, i.e.:

$$p_{lw}^{(0)} = \sigma[A_{lw}^{(1)}] = \frac{1}{1 + e^{-A_{lw}^{(1)}}}, \quad p_{lw}^{(1)} = \sigma[A_{lw}^{(2)}] = \frac{1}{1 + e^{-A_{lw}^{(2)}}} \tag{14}$$

The non-negativity constraints on s_{lw} and σ_{lw} are similarly easy to achieve by passing the activations through the exponential function:

$$s_{lw} = \exp[A_{lw}^{(4)}], \quad \sigma_{lw} = \exp[A_{lw}^{(6)}] \tag{15}$$

More challenging, however, is enforcing the constraints involving the shape parameter of the GP distribution, ξ_{lw}. Recall that there are two constraints on the GP distribution shape parameter: (i) the constraint specified in (13), and (ii) $\xi_{lw} < 1$ to ensure that the mean of the GP distribution, and entire mixture model, is well-defined.

Our approach for ensuring both constraints are satisfied proceeds in three steps. First, a base GP constrainer function is applied to ensure that ξ satisfies constraint (13). Next, a shifted softplus function is used to ensure that the GP distribution shape parameter $\xi_{lw} < 1$. Finally, a gated thresholding function will be applied to ensure that the base GP constrainer and shifted softplus function work appropriately together so that both constraints involving the shape parameter are simultaneously satisfied. We discuss each of these steps to enforce the shape parameter constraint in order.

As mentioned above, the base GP constrainer will ensure that constraint (13) is satisfied. Let $A_{lw}^{(5)}$ and $A_{lw}^{(6)}$ be the unconstrained neural network activations corresponding to the GP distribution parameters ξ_{lw} and σ_{lw}, and let m be the supremum of Y_{lw}. Note that $\sigma_{lw} = \exp(A_{lw}^{(6)})$ as in (15) and define c_ξ to be the base GP constrainer function as follows:

$$\hat{\xi}_{lw} = c_\xi[A_{lw}^{(5)}, A_{lw}^{(6)}] = [\exp(A_{lw}^{(5)}) - 1] \cdot \exp(A_{lw}^{(6)})/(m + \epsilon) \quad (16)$$

The initial output of the base GP constrainer is denoted as $\hat{\xi}_{lw}$ rather than ξ_{lw} to indicate that its output must be further constrained to ensure that the second constraint (i.e. $\xi_{lw} < 1$) is satisfied.

The second constraint $\xi_{lw} < 1$ will be enforced using the shifted softplus function and the gated thresholding function. The shifted softplus function is defined as:

$$S(\hat{\xi}_{lw}) = (1 - \epsilon) - \frac{1}{\beta} \log\left[1 + \exp[(1 - \epsilon - \hat{\xi}_{lw}) \cdot \beta]\right] \quad (17)$$

where β is a hyperparameter (set to 10 for this work), and ϵ is a small positive value (set to 0.05 for this work). The shifted softplus function is a shifted and rotated version of the softplus function. One may verify that $\lim_{\hat{\xi}_{lw} \to \infty} S(\hat{\xi}_{lw}) = (1-\epsilon)$ and $\lim_{\hat{\xi}_{lw} \to -\infty} S(\hat{\xi}_{lw}) = \hat{\xi}_{lw}$. Note that the general outcome of applying the shifted softplus function is to reduce the value of its input so that $S(\hat{\xi}_{lw}) < \hat{\xi}_{lw}$. When $\hat{\xi}_{lw} > 0$ this is no problem since the only constraint ξ_{lw} needs to satisfy is $\xi_{lw} < 1$, but when $\hat{\xi}_{lw} < 0$, this may result in a situation where constraint (13) now becomes violated This is avoided using the gated thresholding function T, defined as:

$$T(\hat{\xi}_{lw}) = v(\hat{\xi}_{lw}) \cdot S(\hat{\xi}_{lw}) + (1 - v(\hat{\xi}_{lw})) \cdot \hat{\xi}_{lw} \quad (18)$$

where

$$v(\hat{\xi}_{lw}) = \begin{cases} 0 & \hat{\xi}_{lw} < 0 \\ \hat{\xi}_{lw}/(1-\epsilon) & 0 < \hat{\xi}_{lw} < 1 - \epsilon \\ 1 & 1 - \epsilon < \hat{\xi}_{lw}. \end{cases} \quad (19)$$

The basic idea of the gated thresholding function T is that when its input $\hat{\xi}$ is less than 0, its input will be returned unchanged. However, when the input $\hat{\xi}$ is greater than $1-\epsilon$, the shifted softplus function is used to reduce its value to be less than 1. When $\hat{\xi}$ is between 0 and $1-\epsilon$, it will smoothly interpolate between the identity function and shifted softplus function to ensure continuity. This results in a function that will constrain $\xi_{lw} < 1$ while also ensuring its output satisfies the GP constraints as long as the input does. Thus, the output of the constraint enforcement module consists of the following estimates of the mixture model distribution parameters:

$$\begin{aligned} \theta_{lw} &= (p_{lw}^{(0)}, p_{lw}^{(1)}, \mu_{lw}, s_{lw}, \xi_{lw}, \sigma_{lw}) \\ &= \Big(\sigma(A_{lw}^{(1)}),\ \sigma(A_{lw}^{(2)}),\ A_{lw}^{(3)},\ \exp(A_{lw}^{(4)}), \\ &\qquad T\Big[c_\xi(A_{lw}^{(5)},\ A_{lw}^{(6)})\Big],\ \exp(A_{lw}^{(6)})\Big) \end{aligned} \quad (20)$$

4.4 Training

DEMM is trained to minimize the following loss function

$$\mathcal{L} = (1 - \lambda)\mathcal{L}_{\text{NLL}} + \lambda\mathcal{L}_{\text{RMSE}} \quad (21)$$

where $\mathcal{L}_{\text{RMSE}} = \sqrt{\frac{1}{LW}\sum_{l,w}(Y_{lw} - \hat{Y}_{lw})^2}$ and $\mathcal{L}_{\text{NLL}}$ denotes the negative log-likelihood function given in (11). λ is a hyperparameter representing the tradeoff between minimizing the negative log-likelihood and root mean squared error loss. One challenge when training the model is choosing the appropriate value for the threshold U_{lw} at each location and time window. To provide more flexibility and allow users to chose any reasonable threshold at test time, during training U_{lw} is sampled uniformly at random from the interval (0.5, 0.95). In principle the range from which the threshold is randomly selected could be extended. This ensures that at test time any threshold from this interval is usable without retraining the model. The *DEMM* framework is trained using Adam [14].

5 EXPERIMENTAL EVALUATION

5.1 Data

We evaluate our model on a real world precipitation dataset drawn from two sources. Predictors are precipitation forecasts from the SubX project.[1] Specifically, an 11-member ensemble of daily precipitation forecasts is generated every week by a numerical model for each location for the next 35 days (i.e $X_{lwi} \in \mathbb{R}^{11}, i \in \{1, 2, \cdots, 35\}$). We compute the rolling 3-day average of each ensemble member. Our target is observed precipitation from NLDAS-2[2]—specifically, the average observed precipitation at each location 10-12 days in advance. We limit our experiments to the continental United States at a 1 degree resolution over 1999-2020. The predictors are log transformed and standardized.

5.2 Models

We consider the following models in our experiments. A comparison is presented in Table 1.

(i) ***DEMM*:** The proposed model described in Section 4 and depicted in Figure 2, trained with a variable threshold U_{lw} defining extremes.
(ii) ***DEMM-F*:** The proposed model, trained with a fixed threshold U_{lw} defining extremes.
(iii) **Hurdle [9]:** An ablation of the proposed model, keeping the 3D CNN module, but omitting the GP component of the mixture distribution. This is equivalent to modeling precipitation with a standard zero/nonzero hurdle model.
(iv) **Vandal [24]:** A baseline for spatiotemporal variables with discrete-continuous structure, using a 3D CNN with Monte

[1] http://cola.gmu.edu/subx/
[2] https://ldas.gsfc.nasa.gov/nldas/v2/forcing

Table 1: A comparison of the models evaluated in our experiments. Columns S, T, Z, and E indicate whether a model is designed to capture spatial, temporal, zero-inflated, and extreme-valued structure, respectively.

	Prediction		Structure			
Model	**Point**	**Distribution**	**S**	**T**	**Z**	**E**
DEMM	✓	✓	✓	✓	✓	✓
DEMM-F	✓	✓	✓	✓	✓	✓
Hurdle [9]	✓	✓	✓	✓	✓	✗
Vandal [24]	✓	✓	✓	✗	✓	✗
Ding [10]	✓	✗	✗	✓	✗	✓
DCNN [12]	✓	✗	✓	✗	✗	✗
Mean	✓	✗	✗	✗	✗	✗

Carlo dropout for uncertainty quantification and distribution prediction.

(v) **Ding [10]:** A baseline for time series prediction with extreme values, using an EVT-motivated loss function and memory module to capture extremes.
(vi) **DCNN [12]:** A deterministic 3D CNN, trained to minimize RMSE of predicted precipitation (without EVT and zero-inflation).
(vii) **Mean:** An average of the ensemble member predictors over days 10-12.

5.3 Experimental Setup

Hyperparameters were selected using grid search. Learning rates varied in the range from 1×10^{-4} to 1×10^{-2}; hidden dimension varied in the range from 10 to 40; tradeoff λ between NLL and MSE varied in the range from 0.7 to 0.9. The optimal hyperparameters for *DEMM* were found to be a learning rate of 1e-3, a hidden dimension of 30, and a λ of 0.9. The 3D CNN in *DEMM* was fixed with 4 layers and the local FCN was fixed with 3 layers. All models were trained for 200 epochs with checkpoints saved at lowest validation loss, then reloaded at test time. All prediction windows were randomly assigned to the train, validation, or test set based on a random seed, with a total of 104 prediction windows (2 years' worth) being assigned to the test set, 104 to the validation set, and the remaining 731 to the test set. A total of five random train-validation-test splits were used to compute averages and standard deviations in each metric. We consider the following evaluation metrics in our experiments.

(i) **RMSE:** Root mean squared error of each model's point prediction, characterizing the average residual magnitude.
(ii) **NLL:** Negative log likelihood of test samples given each model's predicted conditional distribution, characterizing the fidelity of a predicted distribution's center and spread beyond RMSE.
(iii) **Accuracy:** Test precipitation samples are assigned to one of three classes: zero rainfall, moderate rainfall, and extreme rainfall, where extreme rainfall is rainfall that exceeds U_{lw}. For each model, samples can be assigned class probabilities using that model's CDF then assigned to the class with the highest probability, yielding classification accuracy.
(iv) **F1 Macro/Micro:** A macro/micro-averaged F1 score characterizing each model's ability to distinguish between zeros, moderates, and extremes.
(v) **AUC OVO/OVR** Area under the precision-recall curve of one-vs-one/one-vs-rest classification characterizing each model's ability to distinguish between zeros, moderates, and extremes.

5.4 Experimental Results

In addition to characterizing the overall performance of *DEMM*, the experiments were designed to:

(i) Compare the performance of *DEMM* to state-of-the-art baselines and ablations of the full proposed model.
(ii) Compare the performance of *DEMM* against *DEMM-F* to understand the effect of using a variable versus fixed threshold used to define extreme values during training.
(iii) Characterize the spatial locations where the *DEMM* outperforms the ensemble mean.
(iv) Evaluate the ability of the *DEMM* to predict the frequency and timing of extreme events.

5.4.1 Performance Against Baselines. Table 2 compares the overall predictive performance of *DEMM* against the previously discussed baseline methods. For evaluation purposes, the excess threshold U_{lw} was set to the global 0.6 quantile value. We consider *DEMM* trained with varying thresholds as described in Section 4.4 and *DEMM-F*, with a fixed excess threshold at the global 0.6 quantile value of precipitation. In the former case, the results are reported when the threshold is set to the 0.6 quantile at test time. Our experimental results show that both versions of *DEMM* (fixed and variable threshold) outperform the ensemble mean in terms of their negative log likelihood, MSE, and F1 score, and rank competitively among other state-of-the-art baselines. The *DEMM* achieves lower MSE than the ensemble mean. Because the ensemble mean is expected to be a strong baseline, this demonstrates that the *DEMM*'s ability to make accurate point predictions was not strongly inhibited by simultaneously predicting the conditional distribution. The fact that *DEMM* outperforms the hurdle model across various metrics demonstrates the value of incorporating EVT.

Further results are presented in Tables 3 and 4, subsetting results by the component of the target distribution to which each Y_{lw} belonged: observations for which $Y_{lw} = 0$ fall into the Zero column, observations with $0 < Y_{lw} < U_{lw}$ fall into the Moderate column, and observations for which $Y_{lw} > U_{lw}$ fall into the Extreme column. We observe that *DEMM* achieves the lowest RMSE and NLL metrics on the extreme component of our data, suggesting the explicit incorporation of EVT through the generalized Pareto distribution is indeed effective. Similarly, we observe that *DEMM*, *DEMM-F* and the hurdle model all perform well on the zero component of our data, supporting the notion that a discrete-continuous modeling framework better captures zero-inflated data than an assumption of continuity.

5.4.2 Effect of Variable Threshold. A key novelty of *DEMM* is the use of variable thresholds during training to improve generalization at inference-time without the need for retraining. We find that the performance of *DEMM* is not penalized by learning to account for this variable threshold, even beating *DEMM-F* in extreme-valued

Table 2: Results over 5 random test splits of 104 (2 years' worth) prediction windows, denoted $\bar{x} \pm s$ representing sample mean and standard deviation. Gold, silver, and bronze entries denote the best, second, and third result for each metric.

Model	RMSE ↓	NLL ↓	Accuracy ↑	F1 Macro ↑	F1 Micro ↑	AUC OVO ↑	AUC OVR ↑
DEMM	**3.312 ± 0.142**	**1.140 ± 0.067**	0.267 ± 0.003	0.186 ± 0.003	0.267 ± 0.003	**0.675 ± 0.003**	**0.668 ± 0.003**
DEMM-F	3.304 ± 0.152	**2.179 ± 0.080**	**0.334 ± 0.004**	0.296 ± 0.003	**0.334 ± 0.004**	**0.639 ± 0.002**	**0.627 ± 0.002**
Hurdle [9]	3.935 ± 0.235	2.251 ± 0.054	0.223 ± 0.005	0.232 ± 0.004	0.223 ± 0.005	0.641 ± 0.005	0.658 ± 0.004
Vandal [24]	6.933 ± 2.434	1.485 ± 0.013	0.389 ± 0.002	**0.308 ± 0.020**	0.389 ± 0.002	0.510 ± 0.007	0.509 ± 0.007
Ding [10]	4.300 ± 0.263	N/A	**0.393 ± 0.001**	0.233 ± 0.004	**0.393 ± 0.001**	0.504 ± 0.002	0.506 ± 0.002
DCNN [12]	**3.257 ± 0.147**	N/A	0.296 ± 0.003	0.257 ± 0.005	0.296 ± 0.003	0.564 ± 0.003	0.556 ± 0.003
Mean	3.891 ± 0.101	N/A	0.307 ± 0.003	**0.279 ± 0.004**	0.307 ± 0.003	0.567 ± 0.003	0.559 ± 0.003

Table 3: RMSE partitioned by class of Y_{l_w} into zero, moderate, and extreme components. Notation equivalent to Table 2.

	RMSE ↓		
Model	Zero	Moderate	Extreme
DEMM	2.321 ± 0.074	2.329 ± 0.069	**5.576 ± 0.363**
DEMM-F	1.927 ± 0.052	2.258 ± 0.049	5.681 ± 0.379
Hurdle [9]	2.123 ± 0.124	1.784 ± 0.082	5.804 ± 0.401
Vandal [24]	6.309 ± 3.980	7.087 ± 3.185	6.077 ± 0.626
Ding [10]	**1.287 ± 0.019**	**1.127 ± 0.013**	6.653 ± 0.426
DCNN [12]	**1.996 ± 0.037**	**2.170 ± 0.034**	**5.667 ± 0.404**
Mean	2.809 ± 0.055	3.297 ± 0.069	5.576 ± 0.372

Table 4: NLL partitioned by class of Y_{l_w} into zero, moderate, and extreme components. Notation equivalent to Table 2.

	NLL ↓		
Model	Zero	Moderate	Extreme
DEMM	**1.634 ± 0.014**	**0.591 ± 0.082**	**2.960 ± 0.047**
DEMM-F	4.289 ± 0.029	1.819 ± 0.085	3.071 ± 0.089
Hurdle [9]	0.815 ± 0.004	**1.198 ± 0.118**	**3.893 ± 0.080**
Vandal [24]	**0.691 ± 0.012**	0.735 ± 0.008	4.197 ± 0.117
Ding [10]	N/A	N/A	N/A
DCNN [12]	N/A	N/A	N/A
Mean	N/A	N/A	N/A

NLL and RMSE. Indeed, Tables 2, 3, and 4 demonstrate that regardless of the chosen metrics, the two methods results in almost identical performance.

5.4.3 Spatial Analysis. Figure 3 shows the spatial distribution of locations where *DEMM* improves over the hurdle ablation and ensemble mean baseline. The average MAE of the *DEMM* at each of location, taken over all samples and all data splits, is subtracted from the corresponding MAE for one of the baselines so that positive values shaded in red (blue) represent locations where *DEMM* outperforms (underperforms) the baseline. It is clear that *DEMM* outperforms the hurdle model at almost all locations. We see that the improvement in absolute error over the ensemble mean baseline

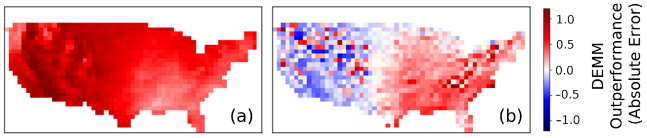

Figure 3: A spatial view comparing the performance of (a) *DEMM* with its hurdle model ablation and (b) *DEMM* with an ensemble mean baseline, as measured by a signed difference in mean absolute error. Red indicates *DEMM* is outperforming; blue indicates *DEMM* is underperforming.

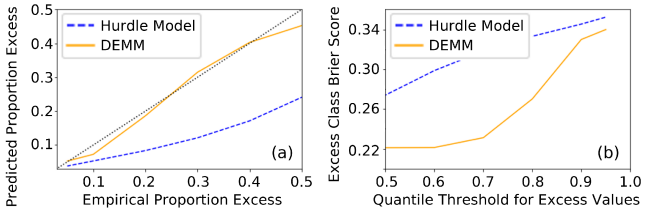

Figure 4: *DEMM* outperforms its hurdle model ablation in predicting the frequency and timing of extreme events. With respect to frequency, *DEMM* predicts a proportion of extreme events aligned with the empirically-observed proportion of extremes (a). With respect to timing, *DEMM* achieves a low Brier score in classifying extreme events (b).

is less consistent; nevertheless, the improvements over the ensemble mean baseline are concentrated at locations in the Eastern U.S. where precipitation values are largest and most variable.

5.4.4 Extreme Event Frequency & Timing. In addition to the previous evaluations, we wish to evaluate whether *DEMM* is able to correctly predict the frequency and timing of extreme events. To examine predictive performance with respect to frequency, in Figure 4(a) we plot predicted vs. empirical frequency of extreme values at varying thresholds. The empirical frequency is the quantile defining the extreme threshold within the model, while the predicted frequency is calculated by computing the probability of an excess value occurring (as described in Section 5.3) averaged across all samples in the test set. We find that the hurdle model consistently under predicts the frequency of extreme values regardless of the quantile

threshold, while the *DEMM* accurately predicts their frequency across all thresholds. This suggests that the *DEMM* is well-suited for predicting the frequency of extreme events regardless of the threshold used to define them.

To examine predictive performance with respect to timing, we compare the Brier score of *DEMM* with the hurdle model using a variety of thresholds to define extreme values in Figure 4(b). The Brier score is a classification metric common in meteorology, with a lower Brier score representing better predictive performance. Given the set of binary class labels for every observed precipitation value, $\{Y_i \mid i \in \{1, \cdots, n\}\}$, which represents whether or not each sample is an extreme value, and the set of predicted probability of excess for each sample, $\{\hat{Y}_i \mid i \in \{1, \cdots, n\}\}$, then the Brier score can be computed as $B = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$. The results shown in Figure 4(b) suggest that *DEMM* consistently outperforms the hurdle model with respect to extreme event timing regardless of the threshold chosen to define extreme events.

6 CONCLUSION

In this work, we propose *DEMM*, a novel deep learning framework for predicting spatiotemporal variables with zero-inflated and heavy-tailed structure. The proposed framework is built upon a mixture model incorporating EVT to model the distribution of extreme events while accurately making point predictions. The framework employs a set of novel reparameterization techniques to ensure that neural network outputs satisfy the constraints placed on the parameters of the mixture model, including a constraint on the GP distribution shape parameter required for computing the mean of the mixture model. The proposed framework also allows the excess threshold to be an input to the model, thus providing flexibility for the user to alter the threshold at inference-time without retraining. Our experiments on a real world precipitation dataset illustrate that *DEMM* is competitive against existing deep learning frameworks for spatiotemporal distribution and point prediction, exhibiting a strong advantage in forecasting the intensity, frequency, and timing of extreme events.

ACKNOWLEDGEMENTS

This research is partially supported by the National Science Foundation under grant IIS-2006633. Any use of trade, firm, or product names is for descriptive purposes only and does not imply endorsement by the U.S. Government.

REFERENCES

[1] Zubin Abraham and Pang-Ning Tan. 2009. A Semi-supervised Framework for Simultaneous Classification and Regression of Zero-Inflated Time Series Data with Application to Precipitation Prediction. In *Proceedings of ICDM Workshop on Spatial and Spatio-temporal Data Mining (STDM '09)*. 644–649.

[2] Zubin Abraham and Pang-Ning Tan. 2010. An Integrated Framework for Simultaneous Classification and Regression of Time-Series Data. In *Proceedings of SIAM International Conference on Data Mining*. 653–664.

[3] Deepak K. Agarwal, Alan E. Gelfand, and Steven Citron-Pousty. 2002. Zero-inflated models with application to spatial count data. *Environmental and Ecological Statistics* 9, 4 (2002), 341–355.

[4] Siddharth Bhatia, Arjit Jain, and Bryan Hooi. 2021. ExGAN: adversarial generation of extreme samples. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence*. 6750–6758.

[5] Christopher M. Bishop. 1994. Mixture density networks. *Neural Computing Research Group, Aston University* (1994).

[6] Gustau Camps-Valls, Devis Tuia, Xiao Xiang Zhu, and Markus Reichstein. 2021. *Deep learning for the Earth Sciences: A comprehensive approach to remote sensing, climate science and geosciences*. John Wiley & Sons.

[7] Rafaela Castro, Yania M Souto, Eduardo Ogasawara, Fabio Porto, and Eduardo Bezerra. 2021. STconvS2S: Spatiotemporal convolutional sequence to sequence network for weather forecasting. *Neurocomputing* 426 (2021), 285–298.

[8] Stuart Coles. 2001. *An Introduction to Statistical Modeling of Extreme Values*. Springer.

[9] John G Cragg. 1971. Some statistical models for limited dependent variables with application to the demand for durable goods. *Econometrica: Journal of the Econometric Society* (1971), 829–844.

[10] Daizong Ding, Mi Zhang, Xudong Pan, Min Yang, and Xiangnan He. 2019. Modeling Extreme Events in Time Series Prediction. In *Proceedings of the Twenty-Fifth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '19)*. 1114–1122.

[11] Asadullah Hill Galib, Andrew McDonald, Tyler Wilson, Lifeng Luo, and Pang-Ning Tan. 2022. DeepExtrema: A Deep Learning Approach for Forecasting Block Maxima in Time Series Data. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI '22)*.

[12] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 2012. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 1 (2012), 221–231.

[13] Anastasiia Kabeshova, Yiyang Yu, Bertrand Lukacs, Emmanuel Bacry, and Stéphane Gaïffas. 2020. ZiMM: A deep learning model for long term and blurry relapses with non-clinical claims data. *J. Biomed. Informatics* 110 (2020), 103531.

[14] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *Proceedings of the Third International Conference on Learning Representations (ICLR '15)*.

[15] Shufeng Kong, Junwen Bai, Jae Hee Lee, Di Chen, Andrew Allyn, Michelle Stuart, Malin Pinsky, Katherine Mills, and Carla P Gomes. 2020. Deep hurdle networks for zero-inflated multi-target regression: Application to multiple species abundance estimation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI '20)*.

[16] Diane Lambert. 1992. Zero-inflated Poisson regression, with an application to defects in manufacturing. *Technometrics* 34, 1 (1992), 1–14.

[17] Andrew McDonald, Pang-Ning Tan, and Lifeng Luo. 2022. COMET Flows: Towards Generative Modeling of Multivariate Extremes and Tail Dependence. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence (IJCAI '22)*.

[18] Evan Racah, Christopher Beckham, Tegan Maharaj, Samira Ebrahimi Kahou, Prabhat, and Christopher Pal. 2017. ExtremeWeather: A large-scale climate dataset for semi-supervised detection, localization, and understanding of extreme weather events. In *Proceedings of the Thirty-First Conference on Neural Information Processing Systems (NeurIPS '17)*.

[19] Stephan Rasp, Peter D Dueben, Sebastian Scher, Jonathan A Weyn, Soukayna Mouatadid, and Nils Thuerey. 2020. WeatherBench: a benchmark data set for data-driven weather forecasting. *Journal of Advances in Modeling Earth Systems* 12, 11 (2020).

[20] Suman Ravuri, Karel Lenc, Matthew Willson, Dmitry Kangin, Remi Lam, Piotr Mirowski, Megan Fitzsimons, Maria Athanassiadou, Sheleem Kashem, Sam Madge, et al. 2021. Skilful precipitation nowcasting using deep generative models of radar. *Nature* 597, 7878 (2021), 672–677.

[21] Laura Serra, Marc Saez, Pablo Juan, Diego Varga, and Jorge Mateu. 2014. A spatio-temporal Poisson hurdle point process to model wildfires. *Stochastic environmental research and risk assessment* 28, 7 (2014), 1671–1684.

[22] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3D convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*. 4489–4497.

[23] Du Tran, Jamie Ray, Zheng Shou, Shih-Fu Chang, and Manohar Paluri. 2017. Convnet architecture search for spatiotemporal feature learning. *arXiv:1708.05038* (2017).

[24] Thomas Vandal, Evan Kodra, Jennifer Dy, Sangram Ganguly, Ramakrishna Nemani, and Auroop R. Ganguly. 2018. Quantifying uncertainty in discrete-continuous and skewed data with Bayesian deep learning. In *Proceedings of the Twenty-Fourth ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18)*. 2377–2386.

[25] Cinzia Viroli and Geoffrey J McLachlan. 2019. Deep gaussian mixture models. *Statistics and Computing* 29, 1 (2019), 43–51.

[26] Tsui-Wei Weng, Huan Zhang, Pin-Yu Chen, Jinfeng Yi, Dong Su, Yupeng Gao, Cho-Jui Hsieh, and Luca Daniel. 2018. Evaluating the robustness of neural networks: An extreme value theory approach. In *Proceedings of the Sixth International Conference on Learning Representations (ICLR '18)*.

[27] Xiaolei Yu, Zhibin Zhao, Xingwu Zhang, Qiyang Zhang, Yilong Liu, Chuang Sun, and Xuefeng Chen. 2021. Deep Learning-Based Open Set Fault Diagnosis by Extreme Value Theory. *IEEE Transactions on Industrial Informatics* (2021).

[28] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *Proceedings of the Sixth International Conference on Learning Representations (ICLR '18)*.