Detecting Anomalous Event Sequences with Temporal Point Processes

Oleksandr Shchur [1 2] **Ali Caner Türkmen** [2] **Tim Januschowski** [2] **Jan Gasthaus** [2] **Stephan Günnemann** [1]

Abstract

Automatically detecting anomalies in event data can provide substantial value in domains such as healthcare, DevOps, and information security. In this paper, we frame the problem of detecting anomalous continuous-time event sequences as out-of-distribution (OoD) detection for temporal point processes (TPPs). We show how this problem can be approached using tools from the goodness-of-fit (GoF) testing literature. Our experiments demonstrate that the proposed method excels at detecting anomalies in both synthetic and real-world data.

1. Introduction

Event data is abundant in the real world and is encountered in various important applications. For example, transactions in financial systems, server logs, and user activity traces can all naturally be represented as variable-length continuous-time event sequences. Detecting anomalies in such data can provide immense industrial value. For example, abnormal entries in system logs may correspond to unnoticed server failures, atypical user activity in computer networks may correspond to intrusions, and irregular patterns in financial systems may correspond to fraud. Manual inspection and handcrafted rules are both infeasible for this task due to massive volume of the data and changing trends (He et al., 2016). Ideally, we would like to have an adaptive system that can learn the normal behavior from the data in an unsupervised way and automatically detect abnormal event sequences.

In this paper, we formulate the problem of detecting anomalous event sequences as an instance of out-of-distribution (OoD) detection. Specifically, we propose an approach for anomaly detection with temporal point processes (TPPs) based on the connection between OoD detection and GoF testing. We also point out that OoD detection is not equivalent to GoF testing, which advances the discussion on this topic started by Nalisnick et al. (2019).

[1]Technical University of Munich [2]Amazon Research. Correspondence to: Oleksandr Shchur <shchur@in.tum.de>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning., Copyright 2021 by the author(s).

2. Problem formulation

Background. A temporal point process (TPP) (Daley & Vere-Jones, 2003), denoted as $\mathbb{P}$, defines a probability distribution over variable-length event sequences in an interval $[0, T]$. A TPP realization X consists of strictly increasing arrival times $(t_1, \dots, t_N)$, where N, the number of events, is itself a random variable. A TPP is characterized by its conditional intensity function $\lambda^*(t) := \lambda(t|\mathcal{H}_t)$ that is equal to the rate of arrival of new events given the history $\mathcal{H}_t = \{t_j : t_j < t\}$. Equivalently, a TPP can be specified with the integrated intensity function (a.k.a. the compensator) $\Lambda^*(t) = \int_0^t \lambda^*(u)du$. For an overview of TPPs, see, e.g., Rasmussen (2018) or Shchur et al. (2021).

Out-of-distribution (OoD) detection. We formulate the problem of detecting anomalous event sequences as OoD detection (Liang et al., 2018). Namely, we assume that we are given a large set of training sequences $\mathcal{D}_{\text{train}} = \{X_1, \dots, X_M\}$ that were sampled i.i.d. from some *unknown* distribution $\mathbb{P}_{\text{data}}$ over a domain $\mathcal{X}$. At test time, we need to determine whether a new sequence X was also drawn from $\mathbb{P}_{\text{data}}$ (i.e., X is in-distribution) or from another distribution $\mathbb{Q} \neq \mathbb{P}_{\text{data}}$ (i.e., X is out-of-distribution or anomalous). We can phrase this problem as a null hypothesis test:

$$\begin{aligned} &H_0 \colon X \sim \mathbb{P}_{\text{data}} \\ &H_1 \colon X \sim \mathbb{Q} \quad \text{for some} \quad \mathbb{Q} \neq \mathbb{P}_{\text{data}}. \end{aligned} \tag{1}$$

Although we consider the case where X is an event sequence sampled from an unknown TPP, the rest of the discussion in Section 2 applies to other data types, such as images.

Goodness-of-fit (GoF) testing. First, we observe that the problem of OoD detection is closely related to the problem of GoF testing (D'Agostino, 1986). We now outline the setup and approaches for GoF testing, and then describe how these can be applied to OoD detection. The goal of a GoF test to determine whether a random element X follows a *known* distribution $\mathbb{P}_{\text{model}}$[1]

$$\begin{aligned} &H_0 \colon X \sim \mathbb{P}_{\text{model}} \\ &H_1 \colon X \sim \mathbb{Q} \quad \text{for some} \quad \mathbb{Q} \neq \mathbb{P}_{\text{model}}. \end{aligned} \tag{2}$$

[1]We test a single realization X, as is common in TPP literature (Brown et al., 2002). Note that this differs from works on *univariate* GoF testing that consider multiple realizations, i.e., $H_0 \colon X_1, \dots, X_M \overset{\text{i.i.d.}}{\sim} \mathbb{P}_{\text{model}}$.

We can perform such a test by defining a test statistic $s(X)$, where $s\colon \mathcal{X} \to \mathbb{R}$ (Fisher, 1936). For this, we compute the (two-sided) p-value for an observed realization x of X as[2]

$$p_s(x) = 2 \times \min\{ \Pr(s(X) \leq s(x)|H_0), \\ 1 - \Pr(s(X) \leq s(x)|H_0)\}. \tag{3}$$

The factor 2 accounts for the fact that the test is two-sided. We reject the null hypothesis (i.e., conclude that X doesn't follow $\mathbb{P}_{\text{model}}$) if the p-value is below some predefined confidence level α. Note that computing the p-value requires evaluating the cumulative distribution function (CDF) of the sampling distribution, i.e., the distribution test statistic $s(X)$ under the null hypothesis H_0.

GoF testing vs. OoD detection. The two hypothesis tests (Equations 1 and 2) appear similar—in both cases the goal is to determine whether X follows a certain distribution $\mathbb{P}$ and no assumptions are made about the alternative $\mathbb{Q}$. This means that we can perform OoD detection using the procedure described above, that is, by defining a test statistic $s(X)$ and computing the respective p-value (Equation 3). However, in case of GoF testing (Equation 2), the distribution $\mathbb{P}_{\text{model}}$ is known. Therefore, we can often analytically compute or approximate the CDF of $s(X)|X \sim \mathbb{P}_{\text{model}}$, and thus the p-value. In contrast, in an OoD detection hypothesis test (Equation 1), we make no assumptions about $\mathbb{P}_{\text{data}}$ and only have access to samples $\mathcal{D}_{\text{train}}$ that were drawn from this distribution. For this reason, we cannot compute the CDF of $s(X)|X \sim \mathbb{P}_{\text{data}}$ analytically. Instead, we can approximate the p-value using the empirical distribution function (EDF) of the test statistic $s(X)$ on $\mathcal{D}_{\text{train}}$ (North et al., 2002).

3. Test statistic for TPPs

We now turn to the literature on GoF testing for TPPs to find test statistics for our OoD detection approach. Many popular GoF tests for TPPs are based on the following result (Ogata, 1988; Brown et al., 2002).

Theorem 1 (Random time change theorem)**.** *A sequence $X = (t_1, \dots, t_N)$ is distributed according to a TPP with compensator Λ^* on the interval $[0, T]$ if and only if the sequence $Z = (\Lambda^*(t_1), \dots, \Lambda^*(t_N))$ is distributed according to the standard Poisson process on $[0, \Lambda^*(T)]$.*

Intuitively, Theorem 1 can be viewed as a TPP analogue of how the CDF of an arbitrary random variable over $\mathbb{R}$ transforms its realizations into samples from $\text{Uniform}([0, 1])$. Similarly, the compensator Λ^* converts a random event sequence X into a realization Z of the standard Poisson process (SPP). Therefore, the problem of GoF testing for an arbitrary TPP reduces to testing whether the transformed sequence Z follows the SPP on $[0, \Lambda^*(T)]$. In other words, we can define a GoF statistic for a TPP with compensator Λ^* by (1) applying the compensator to X to obtain Z and (2) computing one of the existing GoF statistics for the SPP on the transformed sequence. In Appendix C we generalize this to marked TPPs, where each event has a class label.

SPP, i.e., the Poisson process with constant intensity $\lambda^*(t) = 1$, is the most basic TPP one can conceive. Different properties of the SPP can be used to perform a GoF test. For brevity, we denote the transformed arrival times as $Z = (v_1, \dots, v_N) = (\Lambda^*(t_1), \dots, \Lambda^*(t_N))$ and the length of the transformed interval as $V = \Lambda^*(T)$. We can equivalently represent Z by the inter-event times $(w_1, \dots, w_{N+1})$ where $w_i = v_i - v_{i-1}$, assuming $v_0 = 0$ and $v_{N+1} = V$.

KS statistic for the arrival times (Barnard, 1953). Assuming Z follows the SPP, the arrival times $v_1, \dots, v_N$ must be i.i.d. random variables drawn from $\text{Uniform}([0, V])$. It's possible to check how well this assumption holds using the Kolmogorov–Smirnov (KS) statistic. For this, we compare the empirical CDF of the arrival times $\hat{F}_{\text{arr}}(u)$ with $F_{\text{arr}}(u) = u/V$, the CDF of the $\text{Uniform}([0, V])$ distribution. The difference between the two CDFs is quantified using the *KS statistic on the arrival times (KS arrival)*

$$\kappa_{\text{arr}}(Z) = \sqrt{N} \cdot \sup_{u \in [0,V]} |\hat{F}_{\text{arr}}(u) - F_{\text{arr}}(u)|. \tag{4}$$

Chi-squared statistic (Lewis, 1965). The uniformity of the arrival times is can also be quantified using the chi-squared statistic. To compute it, we partition the interval $[0, V]$ into B equally-sized buckets of length $L = V/B$. Then, we compare N_b (the observed number of events in bucket b) with L (the expected number of events in each bucket) as

$$\chi^2(Z) = \sum_{b=1}^{B} \frac{(N_b - L)^2}{L} \tag{5}$$

KS statistic for the inter-event times (Cox, 1966). Another popular GoF test for the SPP is based on the fact that under H_0 the inter-event times w_i are distributed according to the $\text{Exponential}(1)$ distribution. The test compares the empirical CDF of the inter-event times $\hat{F}_{\text{int}}(u)$ with $F_{\text{int}}(u) = 1 - \exp(-u)$, the CDF of the exponential distribution with unit rate. This leads to the *KS statistic for the inter-event times (KS inter-event)*

$$\kappa_{\text{int}}(Z) = \sqrt{N} \cdot \sup_{u \in [0,\infty)} |\hat{F}_{\text{int}}(u) - F_{\text{int}}(u)|. \tag{6}$$

Sum-of-squared spacings statistic. Lastly, we introduce the *sum-of-squared spacings (3S) statistic* defined as

$$\psi(Z) = \frac{1}{V} \sum_{i=1}^{N+1} w_i^2 = \frac{1}{V} \sum_{i=1}^{N+1} (v_i - v_{i-1})^2. \tag{7}$$

[2] In the rest of the paper, we will denote both the random element X and its realization x using the same symbol X.

This statistic extends the sum-of-squared-spacings statistic proposed as a test of uniformity for fixed-length samples by Greenwood (1946). Unlike the earlier-mentioned statistics, the distribution of the 3S statistic is not invariant under changes in the event count N. This will lead to more accurate detection of anomalies, as we will see in Section 6. We discuss other properties of the 3S statistic in Appendix B.

4. OoD detection for TPPs

Our idea is to perform the OoD detection hypothesis test (Equation 1) based on the test statistics from the previous section. Recall that computing these statistics requires using the compensator to obtain the transformed sequence Z. However, in an OoD detection test the data-generating TPP $\mathbb{P}_{\text{data}}$ is unknown, so we don't know the corresponding compensator. Instead, we can fit a neural TPP model $\mathbb{P}_{\text{model}}$ (Shchur et al., 2020) to the sequences in $\mathcal{D}_{\text{train}}$ and use the compensator Λ^* of the learned model to compute the statistic $s(X)$. High flexibility of neural TPPs allows these models to more accurately approximate the true compensator. Having defined the statistic, we can use our approach described in Section 2. That is, we approximate the distribution of the statistic $s(X)$ under H_0 (i.e., assuming $X \sim \mathbb{P}_{\text{data}}$) by the EDF of the statistic on $\mathcal{D}_{\text{train}}$. We use this EDF to compute the p-values for the OoD detection hypothesis test and thus detect anomalous sequences. See Appendix C for a pseudocode description of this method.

We highlight that an OoD detection procedure like the one above is *not* equivalent to a GoF test for the learned generative model $\mathbb{P}_{\text{model}}$, as suggested by earlier works (Nalisnick et al., 2019). While the learned model is used to define the test statistic $s(X)$, we compute the p-value for the OoD detection test based on $s(X)|X \sim \mathbb{P}_{\text{data}}$. This is different from the distribution $s(X)|X \sim \mathbb{P}_{\text{model}}$ used in a GoF test, since in general $\mathbb{P}_{\text{model}} \neq \mathbb{P}_{\text{data}}$. Therefore, even if the distribution of a test statistic under the GoF test can be approximated analytically (as, e.g., for the KS statistic (Marsaglia et al., 2003)), we have to use the EDF of the statistic on $\mathcal{D}_{\text{train}}$ for the OoD detection test. Figure 1 visualizes this difference. Here, we fit a TPP model on the in-distribution sequences from the STEAD dataset (Section 6.2). We plot the empirical distribution of the 3S statistic on $\mathcal{D}_{\text{train}}$ (corresponds to $s(X)|X \sim \mathbb{P}_{\text{data}}$, used to compute p-value for Equation 1) and on samples drawn from the model (corresponds to $s(X)|X \sim \mathbb{P}_{\text{model}}$, used to compute p-value for Equation 2).

5. Related work

The approach presented in Section 2 can be seen as a generalization of many existing methods for unsupervised OoD detection. Existing works usually define the test statistic based on the log-likelihood of a generative model fitted to $\mathcal{D}_{\text{train}}$

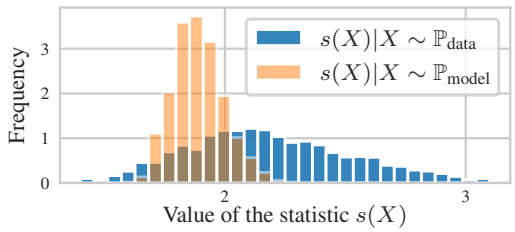

Figure 1. Sampling distribution for the OoD test (blue) and the GoF test (orange). See Section 4 for details.

(Choi et al., 2018; Ren et al., 2019; Nalisnick et al., 2019; Morningstar et al., 2021; Ruff et al., 2021). We don't limit ourselves to log-likelihood and use GoF statistics instead. More importantly, most above papers study OoD detection for image data and none consider variable-length event sequences, which is the focus of our work. Finally, by drawing a clear distinction between OoD detection and GoF testing, we can avoid some of the pitfalls encountered by other works (Nalisnick et al., 2019), as we elaborate in Appendix A.

Our OoD detection procedure is also related to the *rarity* anomaly score (Ferragut et al., 2012). The rarity score can be interpreted as the negative logarithm of a one-sided p-value (Equation 3) of a GoF test that uses the log-likelihood of some *known* model as the test statistic. In contrast, we consider a broader class of statistics and learn the model from data. Existing anomaly detection approaches of TPPs are either limited to Poison processes (Ojeda et al., 2019) or require altering the training procedure (Zhu et al., 2020), while our approach does not have these shortcomings.

6. Experiments

6.1. Detecting anomalies in simulated data

In this section we experimentally evaluate the OoD detection approach proposed in Section 4 on four simulated scenarios. See Appendix C & D for the implementation details.

Data. For each scenario, we define a detectability parameter $\delta \in [0, 1]$. Setting $\delta = 0$ makes $\mathbb{P}_{\text{data}}$ and $\mathbb{Q}$ identical, and increasing δ makes the two distributions more distinct. In SERVER-STOP and SERVER-OVERLOAD we use a Hawkes process (Hawkes, 1971) to simulate communication between 3 hosts in a computer network. In OoD sequences, we change the influence matrix of the Hawkes process to emulate scenarios where a host goes offline (SERVER-STOP), and where a host goes down and the traffic is routed to a different host (SERVER-OVERLOAD). Sequences in LATENCY consist of two event types A and B, where an event of type B happens sometime after each event of type A. In OoD sequences the delay is increased proportionally to the detectability parameter δ. SPIKETRAINS (Stetter et al., 2012) contains sequences of firing times of 50 neurons, each represented by a distinct mark. We generate OoD sequences

Table 1. ROC AUC scores for OoD detection on real-world datasets. Best result in **bold**, results within 2 pp. of the best underlined.

	KS arrival	KS inter-event	Chi-squared	Log-likelihood	3S statistic
LOGS — Packet corruption (1%)	57.4 ± 1.7	62.1 ± 0.9	66.6 ± 1.8	75.9 ± 0.1	**95.5** ± 0.3
LOGS — Packet corruption (10%)	59.2 ± 2.3	97.8 ± 0.6	59.1 ± 2.3	99.0 ± 0.0	**99.4** ± 0.1
LOGS — Packet duplication (1%)	81.1 ± 5.2	82.8 ± 5.0	74.6 ± 6.5	88.1 ± 0.1	**90.9** ± 0.3
LOGS — Packet delay (frontend)	95.6 ± 1.2	98.9 ± 0.4	**99.3** ± 0.1	90.9 ± 0.0	97.6 ± 0.1
LOGS — Packet delay (all services)	**99.8** ± 0.0	94.7 ± 1.1	**99.8** ± 0.0	96.1 ± 0.0	99.6 ± 0.1
STEAD — Anchorage, AK	59.6 ± 0.2	79.7 ± 0.1	67.4 ± 0.2	88.0 ± 0.1	**88.3** ± 0.6
STEAD — Aleutian Islands, AK	53.8 ± 0.5	88.8 ± 0.3	62.2 ± 0.9	97.0 ± 0.0	**99.8** ± 0.0
STEAD — Helmet, CA	59.1 ± 0.9	**98.7** ± 0.0	70.0 ± 0.6	96.9 ± 0.0	92.6 ± 0.3

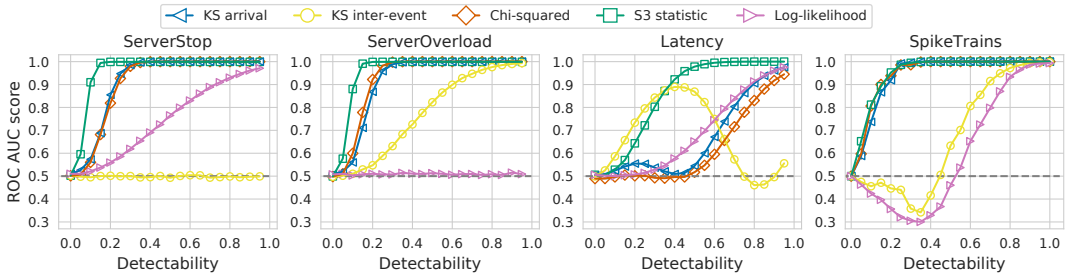

Figure 2. OoD detection on simulated data using different test statistics, measured with with ROC AUC (higher is better).

by shuffling k marks (e.g., switching marks 1 and 2), where higher detectability δ implies more switches k.

Setup. For each scenario and each value of δ, we generate a training set $\mathcal{D}_{\text{train}}$ consisting of in-distribution (ID) sequences, and two test sets $\mathcal{D}_{\text{test}}^{\text{ID}}$ and $\mathcal{D}_{\text{test}}^{\text{OOD}}$ with ID and OoD sequences, respectively. We train an RNN-based neural TPP model (Shchur et al., 2020) on $\mathcal{D}_{\text{train}}$ to define the test statistics. We consider the four statistics described in Section 3: **KS arrival**, **KS inter-event**, **chi-squared**, and **3S** statistics. We additionally consider a two-sided test on the **log-likelihood** of the neural TPP model, similar to Nalisnick et al. (2019). For each test sequence, we compute the p-value (Equation 3) of the OoD detection hypothesis test (Equation 1) using the EDF of the respective statistic on $\mathcal{D}_{\text{train}}$. The p-values of an accurate test must be high for ID sequences and low for OoD sequences—we compute the ROC AUC score on the p-values to quantify the quality of this separation between ID and OoD sequences.

Results are shown in Figure 2. The 3S statistic demonstrates excellent performance in all four scenarios, followed by KS arrival and chi-squared. KS inter-event and log-likelihood statistics completely fail on SERVER-STOP and SERVER-OVERLOAD, respectively, as well as struggle to discriminate OoD sequences in LATENCY and SPIKETRAINS scenarios.

6.2. Detecting anomalies in real-world data

We now apply our methods to detect anomalies in two real-world event sequence datasets. LOGS: We represent server logs generated by Sock Shop microservices (Weave, 2017) as marked event sequences. Sock Shop is a standard testbed for research in microservice applications (Aderaldo et al., 2017) and contains a web application that runs on several containerized services. We generate OoD sequences by injecting various failures (e.g., packet corruption, increased latency) among these microservices using a chaos testing tool Pumba (Ledenev et al., 2016). STEAD (Stanford Earthquake Dataset) (Mousavi et al., 2019) contains seismic measurements of earthquakes. We construct four subsets, each corresponding to a certain geographical location. We treat sequences corresponding the San Mateo, CA region as in-distribution data, and the remaining 3 regions (Anchorage, AK, Aleutian Islands, AK and Helmet, CA) as OoD data.

Results. Table 1 shows the mean and the standard error of the ROC AUC scores computed over 5 initializations. KS arrival and chi-squared achieve low scores in 6 out of 8 scenarios. In contrast, KS inter-event and log-likelihood perform better here than in previous experiments, but still produce poor results on Packet corruption. The 3S statistic is the only method that consistently shows high ROC AUC scores across all scenarios.

7. Conclusion

In this work we have shown how GoF statistics for TPPs can be repurposed for detecting anomalous event sequences. We also proposed the 3S statistic that can be used within the above-mentioned OoD detection approach. We experimentally demonstrated the ability of our method to detect anomalies in a range of simulated and real-world datasets. While our analysis focuses on TPPs, we believe our discussion on similarities and distinctions between GoF testing and OoD detection offers insights to the broader machine learning community.

References

Aderaldo, C. M., Mendonça, N. C., Pahl, C., and Jamshidi, P. Benchmark requirements for microservices architecture research. In *International Workshop on Establishing the Community-Wide Infrastructure for Architecture-Based Software Engineering*, 2017.

Barnard, G. Time intervals between accidents—a note on Maguire, Pearson and Wynn's paper. *Biometrika*, 40(1-2): 212–213, 1953.

Blum, A., Hopcroft, J., and Kannan, R. *Foundations of data science*. 2016.

Brown, E. N., Barbieri, R., Ventura, V., Kass, R. E., and Frank, L. M. The time-rescaling theorem and its application to neural spike train data analysis. *Neural Computation*, 14(2), 2002.

Choi, H., Jang, E., and Alemi, A. A. WAIC, but why? Generative ensembles for robust anomaly detection. *arXiv preprint arXiv:1810.01392*, 2018.

Cox, D. R. The statistical analysis of series of events. *Monographs on Applied Probability and Statistics*, 1966.

D'Agostino, R. B. *Goodness-of-fit-techniques*. 1986.

Daley, D. J. and Vere-Jones, D. *An introduction to the theory of point processes, volume 1: Elementary theory and methods*. 2003.

Ferragut, E. M., Laska, J., and Bridges, R. A. A new, principled approach to anomaly detection. In *International Conference on Machine Learning and Applications*, 2012.

Fisher, R. A. Design of experiments. *British Medical Journal*, 1(3923):554–554, 1936.

Gerhard, F., Haslinger, R., and Pipa, G. Applying the multivariate time-rescaling theorem to neural population models. *Neural computation*, 23(6), 2011.

Greenwood, M. The statistical study of infectious diseases. *Journal of the Royal Statistical Society: Series A*, 109: 85–110, 1946.

Hawkes, A. G. Spectra of some self-exciting and mutually exciting point processes. *Biometrika*, 58(1), 1971.

He, S., Zhu, J., He, P., and Lyu, M. R. Experience report: System log analysis for anomaly detection. In *International Symposium on Software Reliability Engineering*, 2016.

Ledenev, A. et al. Pumba: Chaos testing tool for Docker. https://github.com/alexei-led/pumba, 2016.

Lewis, P. A. W. Some results on tests for Poisson processes. *Biometrika*, 52(1/2), 1965.

Liang, S., Li, Y., and Srikant, R. Enhancing the reliability of out-of-distribution image detection in neural networks. *International Conference on Learning Representations*, 2018.

Marsaglia, G., Tsang, W. W., Wang, J., and Others. Evaluating Kolmogorov's distribution. *Journal of Statistical Software*, 8(18), 2003.

Morningstar, W. R., Ham, C., Gallagher, A. G., Lakshminarayanan, B., Alemi, A. A., and Dillon, J. V. Density of states estimation for out-of-distribution detection. *International Conference on Artificial Intelligence and Statistics*, 2021.

Mousavi, S. M., Sheng, Y., Zhu, W., and Beroza, G. C. STanford EArthquake Dataset (STEAD): A global data set of seismic signals for AI. *IEEE Access*, 2019.

Nalisnick, E., Matsukawa, A., Teh, Y. W., and Lakshminarayanan, B. Detecting out-of-distribution inputs to deep generative models using a test for typicality. *arXiv preprint arXiv:1906.02994*, 2019.

North, B. V., Curtis, D., and Sham, P. C. A note on the calculation of empirical p-values from Monte Carlo procedures. *The American Journal of Human Genetics*, 71 (2), 2002.

Ogata, Y. Statistical models for earthquake occurrences and residual analysis for point processes. *Journal of the American Statistical association*, 83(401), 1988.

Ojeda, C. A. M., Cvejoski, K., Sifa, R., Schuecker, J., and Bauckhage, C. Patterns and outliers in temporal point processes. In *SAI Intelligent Systems Conference*, 2019.

Rasmussen, J. G. Lecture notes: Temporal point processes and the conditional intensity function. *arXiv preprint arXiv:1806.00221*, 2018.

Ren, J., Liu, P. J., Fertig, E., Snoek, J., Poplin, R., Depristo, M., Dillon, J., and Lakshminarayanan, B. Likelihood ratios for out-of-distribution detection. In *Advances in Neural Information Processing Systems*, 2019.

Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., and Müller, K.-R. A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE*, 2021.

Shchur, O., Biloš, M., and Günnemann, S. Intensity-free learning of temporal point processes. In *International Conference on Learning Represenations*, 2020.

Shchur, O., Türkmen, A. C., Januschowski, T., and Günnemann, S. Neural temporal point processes: A review. *International Joint Conference on Artificial Intelligence*, 2021.

Stetter, O., Battaglia, D., Soriano, J., and Geisel, T. Model-free reconstruction of excitatory neuronal connectivity from calcium imaging signals. *PLoS Computational Biology*, 8(8), 2012.

Tao, L., Weber, K. E., Arai, K., and Eden, U. T. A common goodness-of-fit framework for neural population models using marked point process time-rescaling. *Journal of Computational Neuroscience*, 45(2), 2018.

Weave. Sock shop : A microservice demo application. https://github.com/microservices-demo/microservices-demo, 2017.

Zhu, S., Yuchi, H. S., and Xie, Y. Adversarial anomaly detection for marked spatio-temporal streaming data. In *International Conference on Acoustics, Speech and Signal Processing*, 2020.

A. Difference between GoF testing and OoD detection in existing works

The connection between OoD detection and GoF testing was first pointed out by Nalisnick et al. (2019). They proposed to perform a GoF test for a deep generative model to detect OoD instances. However, as we explained in Section 2, these two problems are in fact *not* equivalent. We now demonstrate how this insight allows us to explain and improve upon some results obtained by Nalisnick et al. (2019).

First, we consider the **Gaussian annulus test** for normalizing flow models that was also used by Choi et al. (2018). A normalizing flow model $\mathbb{P}_{\text{model}}$ defines the distribution of a D-dimensional random vector X by specifying a diffeomorphism $f\colon \mathbb{R}^D \to \mathbb{R}^D$, such that $Z = f(X)$ is distributed according to $\mathcal{N}(\mathbf{0}_D, \boldsymbol{I}_D)$, the standard normal distribution. In other words, $f(X)|X \sim \mathbb{P}_{\text{model}}$ follows the standard normal distribution, so any test for the normal distribution can be used to test the GoF of a normalizing flow model. Based on this, Nalisnick et al. (2019) define the following test statistic

$$\begin{aligned}\phi(X) &= \Big| \|f(X)\|_2 - \mathbb{E}_{X\sim\mathbb{P}_{\text{model}}}[\|f(X)\|_2] \Big| \\ &= \Big| \|f(X)\|_2 - \sqrt{D} \Big|.\end{aligned} \tag{8}$$

The idea here is to replace a two-sided test on the statistic $\|f(X)\|_2$ with a one-sided test on the statistic $\phi(X)$ defined above. Since $f(X)|X \sim \mathbb{P}_{\text{model}}$ follows the standard normal distribution, the statistic $\phi(X)|H_0$ will concentrate near 0 (Blum et al., 2016, Theorem 2.9). Therefore, checking if $\phi(X)$ is below a certain threshold ϵ is equivalent to performing the GoF null hypothesis test (Equation 2).

However, the above approach will not work for an OoD detection hypothesis test (Equation 1). If we learn a model $\mathbb{P}_{\text{model}}$ on training instances $\mathcal{D}_{\text{train}}$ that were generated by some distribution $\mathbb{P}_{\text{data}}$, we will in general have $\mathbb{P}_{\text{model}} \neq \mathbb{P}_{\text{data}}$. This implies that $f(X)|X \sim \mathbb{P}_{\text{data}}$ will not follow the standard normal distribution. Therefore, $\mathbb{E}_{X\sim\mathbb{P}_{\text{data}}}[\|f(X)\|_2] \neq \sqrt{D}$ and the distribution of $\|f(X)\|_2$ might not even be symmetric around its mean. This means we cannot replace a two-sided test on $\|f(X)\|_2$ with a one-sided test on $\phi(X)$ when doing OoD detection. A better idea is to directly compute the two-sided p-value for the OoD detection test using the statistic $\|f(X)\|_2$, following our approach in Section 2.

Similarly, for the (single-instance) **typicality test**, the test statistic is defined as

$$\gamma(X) = \Big| \log q(X) - \mathbb{E}_{X\sim\mathbb{P}_{\text{model}}}[\log q(X)] \Big|, \tag{9}$$

where $\log q(X)$ is the log-likelihood of a generative model trained on $\mathcal{D}_{\text{train}}$. This leads to the same problems when trying to apply this statistic for OoD detection as we encountered with the Gaussian annulus test above—the expected value $\mathbb{E}_{X\sim\mathbb{P}_{\text{model}}}[\log q(X)]$ is only suitable for a GoF test. However, in this case Nalisnick et al. (2019) report that they found $\mathbb{E}_{X\sim\mathbb{P}_{\text{data}}}[\log q(X)]$ to work better in practice. By drawing a clear distinction between the OoD detection test and the GoF test we can explain this empirical result. An even better idea is to use the two-sided p-value (Equation 3) instead of Equation 9, since the distribution of the statistic $\log q(X)|X \sim \mathbb{P}_{\text{data}}$ is not guaranteed to be symmetric.

B. Properties of the sum-of-squared spacings (3S) statistic

Intuitively, for a fixed N, the statistic ψ is maximized if the spacings are extremely imbalanced, i.e., if one inter-event time w_i is close to V and the rest are close to zero. Conversely, ψ attains its minimum when the spacings are all equal, that is $w_i = \frac{V}{N+1}$ for all i.

In Figure 3a we visualize the distribution of $\psi|N, V$ for two different values of N. We see that the distribution of ψ depends strongly on N, therefore a GoF test involving ψ will detect if the event count N is atypical for the given SPP. This is in contrast to κ_{arr} and κ_{int}, the distributions of which, by design, are (asymptotically) invariant under N (Figure 3b). Even if one accounts for this effect, e.g., by removing the correction factor $\sqrt{N}$ in Equations 4 and 6, their distributions change only slightly compared to the sum of squared spacings (see Figures 3c and 3d). The chi-squared statistic is similarly invariant to changes in N.

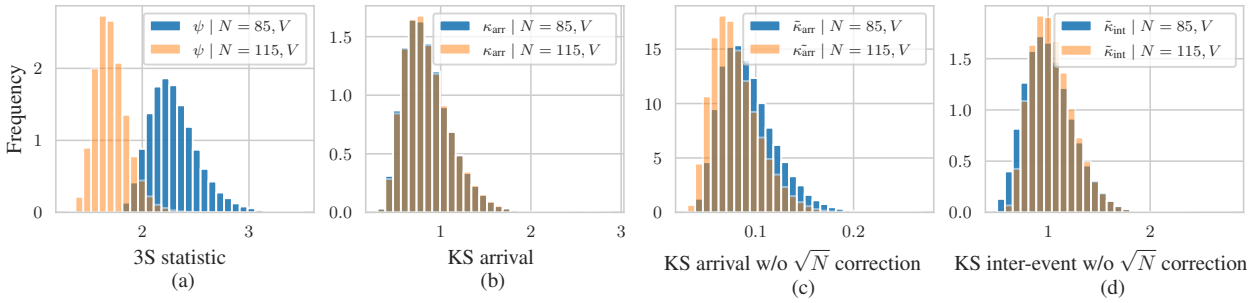

Figure 3. Distribution of different test statistics for the standard Poisson process on $[0, 100]$, conditioned on different event counts N. The 3S statistic allows us to differentiate between different values of N, while the KS statistics are not sensitive to the changes in N.

C. Implementation details

The following code describes the procedure for computing the p-value for the OoD detection test (Equation 1). The code below is for demonstration purposes only, the actual implementation used in our experiments is better optimized.

```
def compute_p_value(x_test, samples, score):
    scores_id = [score(x) for x in samples]
    score_x = score(x_test)
    num_train = len(samples)
    num_above = 0
    for s in scores_id:
        if s > score_x:
            num_above += 1
    num_below = num_train - num_above
    return min(
        (num_below + 1) / (num_train + 1),
        (num_above + 1) / (num_train + 1)
    )
```

The `+1` correction in the numerator and denominator for the p-value computation is done as described by North et al. (2002). Here `samples` is the set of in-distribution sequences $\mathcal{D}_{\text{train}}$ that were generated from $\mathbb{P}_{\text{data}}$. If we instead use realizations drawn from the model $\mathbb{P}_{\text{model}}$ as `samples`, we recover the GoF test (Equation 2) for $\mathbb{P}_{\text{model}}$.

In the snippet above, `score` corresponds to a test statistic $s\colon \mathcal{X} \to \mathbb{R}$. In our experiments, we consider the following choices for s:

1. KS arrival (Equation 4).
2. Chi-squared (Equation 5).
3. KS inter-event (Equation 6).
4. Sum-of-squared spacings (Equation 7).
5. Log-likelihood

$$\log q(X) = \sum_{i=1}^{N} \log \frac{\partial \Lambda^*(t_i)}{\partial t_i} - \Lambda^*(T). \tag{10}$$

All these statistics are computed based on some TPP model with compensator Λ^*. For statistic 1–4, we compute $s(X)$ by first obtaining the transformed sequence $Z = (\Lambda^*(t_1), \dots, \Lambda^*(T))$ and then evaluating the respective SPP statistic on Z. The log-likelihood is directly evaluated based on the model's compensator.

Marked sequences. In a marked sequence $X = \{(t_1, m_1), \dots, (t_N, m_N)\}$ each event is represented by a categorical mark $m_i \in \{1, \dots, K\}$ in addition to the arrival time t_i. A marked TPP model is specified by K compensators $\{\Lambda_1^*, \dots, \Lambda_K^*\}$.

We obtain the transformed sequence Z necessary for statistics 1–4 as follows. Let $\left(t_1^{(k)}, \dots, t_{N_k}^{(k)}\right)$ denote the events of mark k in a given sequence X. For each mark $k \in \{1, \dots, K\}$, we obtain a transformed sequence $Z^{(k)} = \left(\Lambda_k^*(t_1^{(k)}), \dots, \Lambda_k^*(t_{N_k}^{(k)}), \Lambda_k^*(T)\right)$. Then we concatenate the transformed sequences for each mark, thus obtaining a single SPP realization on the interval $[0, \sum_{k=1}^{K} \Lambda_k^*(T)]$. For example, suppose the transformed sequence for the first mark is $Z^{(1)} = (1.0, 2.5, 4.0)$ and for the second mark $Z^{(2)} = (0.5, 3.0)$. Then the concatenated sequence will be $Z = (0.0, 1.0, 2.5, 4.0 + 0.5, 4.0 + 3.0) = (0.0, 1.0, 2.5, 4.5, 7.0)$. Our approach based on concatenating the $Z^(k)$'s is simpler than other methods for combining multiple sequences by Gerhard et al. (2011) & Tao et al. (2018), and we found ours to work well in practice.

The log-likelihood for a marked sequence is computed as

$$\log q(X) = \sum_{k=1}^{K} \sum_{i=1}^{N} \mathbb{1}(m_i = k) \log \frac{\partial \Lambda_k^*(t_i)}{\partial t_i} - \sum_{k=1}^{K} \Lambda_k^*(T). \tag{11}$$

D. Datasets

D.1. Simulated data

SERVER-STOP and SERVER-OVERLOAD: In-distribution sequences for both scenarios are generated by a multivariate Hawkes process with $K = 3$ marks on the interval $[0, 100]$

with following base rates $\boldsymbol{\mu}$ and influence matrix $\boldsymbol{A}$:

$$\boldsymbol{\mu} = \begin{pmatrix} 3 \\ 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{A} = \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

This scenario represents communication between a server (mark 1) and two worker machines (marks 2 and 3) — events for the workers can only be triggered by incoming requests from the server.

In OoD sequences, the structure of the influence matrix is changed at time $t_{\text{stop}} = T(1 - 0.5\delta)$, which represents the time of a failure in the system. For SERVER-STOP, the influence matrix is changed to $\boldsymbol{A}^{\text{stop}}$, and for SERVER-OVERLOAD the influence matrix is changed to $\boldsymbol{A}^{\text{overload}}$.

$$\boldsymbol{A}^{\text{stop}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix} \qquad \boldsymbol{A}^{\text{overload}} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 2 & 0 & 0 \end{pmatrix}$$

The sets $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{test}}^{\text{ID}}$ and $\mathcal{D}_{\text{test}}^{\text{OOD}}$ consist of 1000 sequences each.

LATENCY: Event sequences consist of two marks. ID sequences are generated as follows. Events of the first mark ("the trigger") are generated by a homogeneous Poisson process with rate $\mu = 3$. Events of the the second mark ("the response") are obtained by shifting the arrival times of the first mark by offsets that are sampled i.i.d. from $\text{Normal}(\mu = 1, \sigma = 0.1)$. In OoD sequences, the offsets are instead sampled from $\text{Normal}(\mu = 1 + 0.5\delta, \sigma = 0.1)$. That is, OoD sequences correspond to increased latency between the "trigger" and "response" events. The sets $\mathcal{D}_{\text{train}}$, $\mathcal{D}_{\text{test}}^{\text{ID}}$ and $\mathcal{D}_{\text{test}}^{\text{OOD}}$ consist of 1000 sequences each.

SPIKETRAINS: The original fluorescence data is provided at `www.kaggle.com/c/connectomics`. We extracted the spike times from the fluorescence recordings using the code by `https://github.com/slinderman/pyhawkes/tree/master/data/chalearn`. We dequantized the discrete spike times by adding $\text{Uniform}(-0.5, 0.5)$ noise and selected the first 50 marks.

The original data consists of a single sequence that is 3590 seconds long. We split the long sequence into overlapping windows that are 20 seconds long. We select the first 500 sequences for training (as $\mathcal{D}_{\text{train}}$), and 96 remaining sequences for testing (as $\mathcal{D}_{\text{test}}^{\text{ID}}$). OoD sequences (i.e., $\mathcal{D}_{\text{test}}^{\text{OOD}}$) are obtained by switching $k = \lfloor \delta K \rfloor$ marks. For example, if marks 5 and 10 are switched, all events that correspond to mark 5 in $\mathcal{D}_{\text{test}}^{\text{ID}}$ will be labeled as mark 10 in $\mathcal{D}_{\text{test}}^{\text{OOD}}$, and vice versa.

D.2. Real-world data

LOGS: We ran the Sock Shop microservices testbed (Weave, 2017) on our in-house server. We consider the logs corresponding to the `user` service. There are 4 types of log entries that we model as 4 categorical marks. We use the timestamps of log entries as arrival times of a TPP. We slice the logs into 30-second-long non-overlapping windows, each corresponding to a single TPP realization.

We run the service for ≈14 hours to generate training data, and then for additional ≈5 hours to generate test data. The test data contains 5 types of injected anomalies produced by Pumba (Ledenev et al., 2016). See Table 1 for the list of anomalies. Each anomaly injection lasts 10 minutes. We mark a test sequence as OoD if the system was "attacked" by Pumba during the respective time window. In total, we use 1668 sequences as $\mathcal{D}_{\text{train}}$, 502 sequences as $\mathcal{D}_{\text{test}}^{\text{ID}}$, and 22 sequences as $\mathcal{D}_{\text{test}}^{\text{OOD}}$ for each of the attack scenarios (i.e., 110 OoD sequences in total).

STEAD: The original dataset by Mousavi et al. (2019) contains over 1 million earthquake recordings. We sample 72-hour sub-windows and treat times of earthquake as arrival times of a TPP, as usually done in seismological applications. We treat the sequences as unmarked. We select 4 geographic locations: (1) San Mateo, CA, (2) Anchorage, AK, (3) Aleutian Islands, AK, and (4) Hemet, CA. We group the earthquakes that happen within a 350 km radius (geodesic) around each of the locations, thus obtaining 4 sets of sequences (5000 sequences for each location). We use the sequences corresponding to (1) San Mateo, CA, as in-distribution data, and the remaining 3 locations as OoD data. We use 4000 ID sequences as $\mathcal{D}_{\text{train}}$, 1000 ID sequences and $\mathcal{D}_{\text{test}}^{\text{ID}}$, and 1000 sequences per each remaining location as $\mathcal{D}_{\text{test}}^{\text{OOD}}$.

E. Experimental setup

We train a neural TPP model similar to Shchur et al. (2020). We parametrize the inter-event time distribution with a mixture of 8 Weibull distributions. The marks are conditionally independent of the inter-event times given the context embedding, as in the original model. Mark embedding size is set to 32, and the context embedding (i.e., RNN hidden size) is set to 64 for all experiments.

We optimize the model parameters by maximizing the log-likelihood of the sequences in $\mathcal{D}_{\text{train}}$ (batch size 64) using Adam with learning rate 10^{-3} and clipping the L_2-norm of the gradients to 5. We run the optimization procedure for up to 200 epochs, and perform early stopping if the training loss stops improving for 10 epochs. The p-values are computed according to the procedure described in Appendix C. The results reported in Section 6.1 are averaged over 10 random seeds. In Section 6.2, we train the neural TPP model with 5 different random initializations to compute the average and standard error in Table 1.