Learning Network Traffic Dynamics Using Temporal Point Process

Avirup Saha
IIT Kharagpur
avirupsaha@iitkgp.ac.in

Niloy Ganguly
IIT Kharagpur
niloy@cse.iitkgp.ac.in

Sandip Chakraborty
IIT Kharagpur
sandipc@cse.iitkgp.ac.in

Abir De
MPI-SWS
ade@mpi-sws.org

***Abstract*—Accurate modeling of network traffic has a wide variety of applications. In this paper, we propose Network Transmission Point Process (NTPP), a probabilistic deep machinery that models the traffic characteristics of hosts on a network and effectively forecasts the network traffic patterns, such as load spikes. Existing stochastic models relied on the network traffic being self-similar in nature, thus failing to account for traffic anomalies. These anomalies, such as short-term traffic bursts, are very prevalent in certain modern-day traffic conditions, e.g. datacenter traffic, thus refuting the assumption of self-similarity. Our model is robust to such anomalies since it effectively leverages the self-exciting nature of the bursty network traffic using a temporal point process model.**

On seven diverse datasets collected from the fields of cyber-defense exercises (CDX), website access logs, datacenter traffic, and P2P traffic, NTPP offers a substantial performance boost in predicting network traffic characteristics against several baselines, ranging from forecasting the network traffic volume to detecting traffic spikes. We also demonstrate an application of our model to a caching scenario, showing that it can be used to effectively lower the cache miss rate.

***Index Terms*—time series/data streams, network host behavior modeling, temporal point processes**

I. Introduction

The ever growing demands for new types of network applications and systems have made the network traffic behavior more complex and more unpredictable. For example, in a data-center network, the traffic micro-bursts originated from a sudden surge of an application's popularity [1] and the elephant flows generated during information synchronization among the replicas [2] create temporary load-imbalance over the backbone network. On the other hand, traffic-intensive applications like multimedia streaming and video-conferencing result in a huge traffic-disparity over cellular and mobile networks due to diverse end-user activity patterns [3]. Such traffic-disparity affects the quality of experience (QoE) for the end-user applications [4]. Further, the network traffic variability has increased many-fold with the rapid proliferation of the large scale geo-distributed cloud storage synchronization through Internet Small Computer System Interface (iSCSI) based distributed storage [5] and Internet of Things (IoT) applications [6]. Various kinds of security attacks, like distributed denial-of-service (DDoS), escalate the problem of false-negatives in traffic pattern prediction [7].

With this diverse range of applications, both short-term and long-term traffic-bursts are common for various types of networks; therefore, researchers have explored different techniques to predict traffic bursts with the assumption of burst-periodicity, such as partial predictability of traffic matrix [8], tensor completion approach [9], etc. However, more recently the highly uneven nature of network traffic nullifies such assumptions of traffic-burst periodicity and gives rise to conspicuous traffic disparities and multifractal traffic variabilities, which require separate detection efforts. Examples of such traffic disparities and multifractal traffic variabilities can be seen in sudden spikes in traffic from data centers (micro-bursts) [10] or ISP backbones [11], traffic from multimedia applications (e.g. video streaming) [12], storage synchronization traffic [13], malicious or attack traffic (e.g. DDoS attack on IoT devices) [7], etc. Hence, the requirement is to develop a traffic-event prediction model that can capture such traffic disparities and multifractal traffic variabilities, like traffic bursts, sudden spikes, unexpected jumps in bandwidth usage by a host, etc.

In this work, we aim to integrate disparity and variability detections into network traffic modeling, thus providing a unified model for highly anomalous network traffic. In order to do this, we break down the problem of traffic prediction in terms of transmission characteristics of separate network hosts (such as a data-center server or an end-user device), where we focus on the share of the total network bandwidth utilized by each host, which is termed as its *"dominance"* at a given time. To this end, we propose **N**etwork **T**ransmission **P**oint **P**rocess (NTPP), which is a deep probabilistic machinery based on the mechanism of temporal point processes. NTPP first characterizes the events of bursty traffic generations from a host using a *Recurrent Marked Temporal Point Process* (RMTPP) [14] which incorporates the influence of a host to forward a traffic burst based on the available bandwidth. Furthermore, we model the contention among different hosts using a family of learning to rank templates that rank different hosts on the network at any given time, where the rank of a host is determined by the volume of traffic it generates. Such templates provide various measures to evaluate the relative order of a pair of hosts, induced by their contention process. These measures, together with the underlying packet transmis-

This research was partially funded by the project nos. 6537 and 6720 of "IMPacting Research INnovation and Technology (IMPRINT)", an initiative of DST and MHRD, India as well as the project "Understanding and Predicting Opinion Dynamics In Collective Environment Using Deep Network Representation" funded by CISCO University Research Program Fund.

sion process, ensure the correct ordering among hosts across the entire time-window. To learn the transmission dynamics along with the fluctutations in ranking, we maximize the likelihood of the observed transmission times from given hosts, unified with an additional measure modeled by the learning to rank templates. Such an additional gadget enables our model to forecast unexpected spikes, jumps in bandwidth usage, which otherwise, is extremely difficult to trace (Experiments emphatically establish that).

We evaluate our system against several state-of-the-art baselines on seven real datasets from diverse domains, which are likely to exhibit anomalous traffic. Four of these are obtained from cyber-defense exercises conducted by various organizations, one from website access logs (1998 World Cup webserver), one from data-center traffic, and one from BitTorrent network. We observe that NTPP performs 11% better, on average, than the most competing baseline for forecasting host traffic, while it shows $\sim 25\%$ better prediction accuracy for detecting sudden jumps or spikes in host bandwidth consumption. We have also implemented a downstream caching application using a simulator based on NTPP, and we observe a $\sim 10\%$ reduction in cache miss rate.

Contributions: Summarizing, in this paper, we make the following contributions.

(1) Modeling complex packet transmission process: We devise NTPP, a nonlinear stochastic model for network traffic dynamics with multiple hosts, that accurately captures the presence of aggressive jumps and irregularities in packet transmission process. Furthermore, in contrast to the existing discrete time traffic models (such as [9], [15]), we use the continuous time feature of a temporal point process.

(2) Modeling contention between hosts: Our NTPP proposal leverages the modeling idea of product competition proposed in [16], which connects the rich literature of learning to rank [17] with network traffic modeling.

(3) Forecasting ability: NTPP is not only theoretically grounded, but also practically effective. Our model is able to forecast packet transmission dynamics more effectively than several state-of-the-art baselines. Furthermore, the embedded discriminative module helps to estimate sudden changes in bandwidth consumption on the fly, which is a crucial practical challenge that none of the baselines can even trace.

(4) Downstream application: We demonstrate the application of NTPP to a downstream caching scenario, thus highlighting its practical usefulness. Whereas the existing naive content cache suffers from a high cache miss rate due to bursty traffic, an intelligent content cache supported by our model performs much better by reserving different amounts of memory space for different hosts according to their predicted traffic volumes.

II. Related Work

Historically, a large number of works have focused on the modeling of world wide web traffic from various perspectives [18], using various distribution models such as Poisson, Pareto, Weibull, Markov and Embedded Markov, ON-OFF, and so on. With the growth of the Internet and introduction of a wide variety of web-enabled services, more complex models, such as Markov modulated Poisson process [19], Markov modulated fluid models [20], auto-regressive models [21], partial predictability of traffic matrix [8], tensor completion approach [9], etc. have been proposed. However, such models are able to capture only specific types of network events and cannot be generalized to capture different traffic disparities and variabilities in Internet traffic. In a separate thread, researchers have modeled Internet traffic bursts as a phenomenon which exhibits self-similarity [22]. However, a number of works [23], [24] have also questioned the 'self-similarity' assumption, particularly at the Internet backbone where traffic from multiple sources get multiplexed.

The nature of the Internet traffic has changed drastically with the emergence of diverse domains, like large scale data-centers, IoT based platforms, cellular and mobile networks, information centric networking, etc. Accordingly, various domain-specific models have emerged, such as traffic micro-burst predictions in data centers [15], traffic anomaly detection [25], IoT traffic characterization [26], predicting social network events over the Internet [27], etc. Further, owing to the diverse nature of network traffic under various disparities and variabilities, a number of recent works have explored machine learning based techniques to predict different events, anomalies and inconsistencies in traffic patterns [12], [28], [29]. However, such prediction models are designed for specific networked systems, and lacks generality.

III. Model formulation

In this section, we formulate NTPP, the proposed model (See Figure 1) that captures two major ingredients of network traffic dynamics – (i) the collective packet transmission mechanism, and (ii) the contention between multiple hosts. At the very outset, NTPP is driven by a deep probabilistic machinery based on point process – a special type of stochastic process that naturally captures the mechanism behind sequential packet arrivals. Furthermore, it contains a discriminative module having a family of learning to rank functions [17], specifically engineered to model the inter-host contention process. In the following, we describe them in detail, starting from an overview of temporal point process, and then describe methods for learning and predicting the dynamics.

A. Overview of Temporal Point Process

A temporal point process (TPP) is a stochastic process consisting of discrete events localized in time. That said, formally, any TPP is always associated with a set of timestamps $\mathcal{H}_t = \{t_i < t | i \in \mathbb{Z}^+\}$ indicating a set of events occurred until time t. In the context of network traffic dynamics, we define $\mathcal{H}_{\boldsymbol{H}}(t)$ for a host $\boldsymbol{H}$ as the sequence of packet transmission times by $\boldsymbol{H}$ until time t, i.e. $\mathcal{H}_{\boldsymbol{H}}(t) := \{t_i < t | \boldsymbol{H}$ transmits a packet at time $t_i\}$. $\mathcal{H}_{\boldsymbol{H}}(t)$ can also be called as the history of $\boldsymbol{H}$ until time t. It can also be represented as a

counting process $N_{\boldsymbol{H}}(t) \in \{0\} \cup \mathbb{Z}^+$ which counts the number of transmitted packets of the host $\boldsymbol{H}$ during $[0, t)$. Therefore,

$$N_{\boldsymbol{H}}(t) = \sum_{t_i \in \mathcal{H}_{\boldsymbol{H}}(t)} u(t - t_i) \tag{1}$$

Here $u(t - t_i)$ is a Heaviside step function. We characterize the dynamics of this counting process $N_{\boldsymbol{H}}(t)$ using $\lambda_{\boldsymbol{H}}(t)$ that encodes conditional probability of observing a packet transmission in the infinitesimal time interval $[t, t + dt)$, given the history $\mathcal{H}_{\boldsymbol{H}}(t)$ of packet transmission events until time t:

$$\mathbb{P}(\underbrace{\boldsymbol{H} \text{ transmits a packet in } [t, t + dt)}_{dN_{\boldsymbol{H}}(t)=1} | \mathcal{H}_{\boldsymbol{H}}(t)) = \lambda_{\boldsymbol{H}}(t)\ dt.$$

Here, $dN_{\boldsymbol{H}}(t) = N_{\boldsymbol{H}}(t+dt) - N_{\boldsymbol{H}}(t)$ indicates the number of packet transmission events in the interval $[t, t+dt)$. Assuming packet transmissions are independent, we note that

$$\mathbb{P}(dN_{\boldsymbol{H}}(t) = n | \mathcal{H}_{\boldsymbol{H}}(t)) = O(dt^n) \to 0\ \forall n \geq 2$$

Therefore, a point process always enforces that the arrivals of two packets are asynchronous. So, $dN_{\boldsymbol{H}}(t)$ is either 0 or 1 with probability 1. Hence, it turns out that

$$\mathbb{E}[d\boldsymbol{N}(t) | \mathcal{H}_{\boldsymbol{H}}(t)] = 1.\lambda_{\boldsymbol{H}}(t)dt + 0.(1 - \lambda_{\boldsymbol{H}}(t)dt) = \lambda_{\boldsymbol{H}}(t)dt$$

$$\textit{i.e. } \mathbb{E}[\boldsymbol{N}(T) | \mathcal{H}_{\boldsymbol{H}}(T)] = \int_0^T \lambda_{\boldsymbol{H}}(t)dt \tag{2}$$

Therefore, $\lambda_{\boldsymbol{H}}(t)$ which is used to compute the conditional probability function, also indicates the average rate or intensity of events in any infinitesimal time-window $[t, t + dt)$. So, it is also called the conditional intensity function and may also depend on $\mathcal{H}_{\boldsymbol{H}}(t)$ in general. We note that $\lambda_{\boldsymbol{H}}(t)$ alone specifies the stochastic dynamics of $N_{\boldsymbol{H}}(t)$. Hence, the generative modeling of the packet transmission process, i.e. modeling the generation of t_i's, essentially boils down to an appropriate formulation of $\lambda_{\boldsymbol{H}}(t)$.

B. System model

We leverage the broad framework of the product competition model proposed in [16] in our current task. Figure 1 shows a broad overview of our model, which consists of two parts:

NTPP$_{\text{Gen}}$, the generative module for NTPP: Given a set of hosts $\mathbb{H}$, the transmission times for a host $\boldsymbol{H}$ are governed by the stochastic intensity $\lambda_{\boldsymbol{H}}(t)$ which, in general, depends on the history of all hosts $\boldsymbol{H} \in \mathbb{H}$. That said, we can write $\lambda_{\boldsymbol{H}}(t)$ as:

$$\lambda_{\boldsymbol{H}}(t) = \boldsymbol{\Lambda}(\cup_{\boldsymbol{H} \in \mathbb{H}} \mathcal{H}_{\boldsymbol{H}}(t)), \tag{3}$$

where $\boldsymbol{\Lambda}$ can be any arbitrary nonlinear smooth function, as opposed to parameterized restrictive forms. In this work, we model this function using recurrent neural network (RNN). RNNs are a class of feedforward neural architectures with some auxiliary edges, called recurrent edges, that connect the current signals from the hidden states to the network as the future inputs at the very next time-step. Such recursive units help to create an inbuilt memory, thereby accurately capture the influence of memory from the previous transmissions. At the very outset, our model provides one RNN ($\text{RNN}_{\boldsymbol{H}}$) per each host $\boldsymbol{H}$, which takes previous transmission times $t_i \in \cup_{\boldsymbol{H} \in \mathbb{H}} \mathcal{H}_{\boldsymbol{H}}(t)$ as inputs, and outputs the intensity $\lambda_{\boldsymbol{H}}(t)$ for the next transmission event as shown in Figure 1. In this process, the hidden states of $\text{RNN}_{\boldsymbol{H}}$ embed the history $\mathcal{H}_{\boldsymbol{H}}(t)$ into the vectors $h_k^{\boldsymbol{H}}$ that are recursively constructed from the previous embeddings $h_{k-1}^{\boldsymbol{H}}$ as well as the signals derived from current input timings. Such $h_\bullet^{\boldsymbol{H}}$s are fixed low dimensional representations of the transmission history of host $\boldsymbol{H}$– that grows large as time goes.

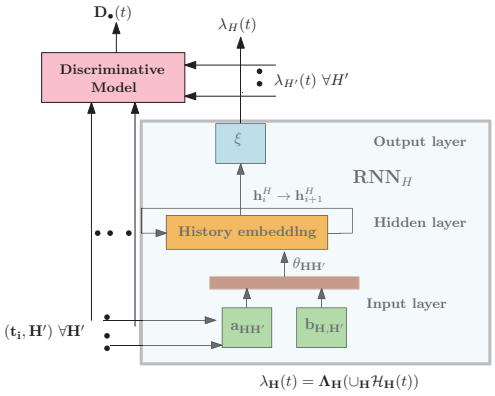

Fig. 1: The neural network architecture of NTPP for a host $\boldsymbol{H}$. At a high level, it shows the generator modeled as RNN and its interaction with discriminator. The current event (t_i) for a host $\boldsymbol{H}'$ is fed to the generator. It undergoes multiple transformations in the input layer and generates embedded signals $\boldsymbol{\theta}_i^{\boldsymbol{H}}$, which are fed into the hidden layer where the states are computed recursively to learn the proper representation to capture the appropriate nonlinear influences of past events. The computed state $\boldsymbol{h}_i^{\boldsymbol{H}}$ is fed to the output layer which computes the conditional intensity $\lambda_{\boldsymbol{H}}(t)$ at time t. The output intensities, along with input timestamps further go as inputs to the discriminative module which captures the ranking dynamics of hosts.

For each host $\boldsymbol{H}$, $\text{RNN}_{\boldsymbol{H}}$ has three layers. The first layer is the *input layer* that takes the packet transmission times for all hosts as inputs, and turns them into suitable signals which are afterwards used as input signals to the next (hidden) layer. That is, once a host $\boldsymbol{H}$ transmits a packet at time t_i, the input layer converts it into a signal $\boldsymbol{\theta}_i^{\boldsymbol{H}} = \boldsymbol{a}_{\boldsymbol{H},\boldsymbol{H}'} t_i + \boldsymbol{b}_{\boldsymbol{H},\boldsymbol{H}'}$. Here $\boldsymbol{a}_{\boldsymbol{H},\boldsymbol{H}'}$ and $\boldsymbol{b}_{\boldsymbol{H},\boldsymbol{H}'}$ are the measures of influence of $\boldsymbol{H}'$ on $\boldsymbol{H}$, thereby enforcing a dependent structure in the traffic flow. The computed vectors $\boldsymbol{\theta}_i^{\boldsymbol{H}}$, as well as the timings t_i are fed into the recurrent unit where the hidden state computes $\boldsymbol{h}_i^{\boldsymbol{H}}$ using the previous embedding $\boldsymbol{h}_{i-1}^{\boldsymbol{H}}$ in the following way:

$$\boldsymbol{h}_i^{\boldsymbol{H}} = \boldsymbol{g}_{\boldsymbol{w}}(l_{\boldsymbol{H}} \boldsymbol{h}_{i-1}^{\boldsymbol{H}} + \underbrace{\boldsymbol{\beta}_{\boldsymbol{H}} e^{-\frac{\omega t_i}{k_i}}}_{\substack{\text{influence of packet} \\ \text{length}}} + \boldsymbol{\gamma}_{\boldsymbol{H}}^T \boldsymbol{\theta}_i^{\boldsymbol{H}}) \tag{4}$$

Here $\boldsymbol{g}_{\boldsymbol{w}}$ is a nonlinear function realized by the neural network with parameters $\boldsymbol{w}$; and $l_{\boldsymbol{H}}$, $\boldsymbol{\beta}_{\boldsymbol{H}}$, and $\boldsymbol{\gamma}_{\boldsymbol{H}}^T$ are parameters to

be learned and act as weights.

Note that k_i in the second term of Eq. 4 measures the length of a packet (in terms of bytes) transmitted at time t_i. This indicates that a host which transmits longer packets has a higher rate of transmission than a host which transmits shorter packets. To this end, a host transmission event, i.e. a (short) long packet, with a (small) high value of k_i, (trims) boosts the values of $\boldsymbol{h}_i^{\boldsymbol{H}}$ that later on (reduces) increases the arrival intensity $\lambda_{\boldsymbol{H}}(t)$ which is modeled in the output layer in the following way:

$$\lambda_{\boldsymbol{H}}(t) = \exp(\boldsymbol{\xi}(\boldsymbol{\omega}^T \boldsymbol{h}_i^{\boldsymbol{H}} + b(t - t_i) + c)) \tag{5}$$

where t_i is the time of the last observed event prior to t. The term with $b(t - t_i)$ makes sure that $\lambda_{\boldsymbol{H}}(t)$ increases with each transmission instance at t_i, thereby capturing the bursty nature of realistic dynamics of the network traffic flow. Moreover, the exponential ensures a positive value of $\lambda_{\boldsymbol{H}}(t)$ that may be otherwise violated during the training process. Note that, $\boldsymbol{\xi}$ is also realized using a neural network. Here $\boldsymbol{\omega}^T$, b and c are learnable parameters and act as weights ($\boldsymbol{\omega}^T$, b) and bias (c).

NTPP$_{\text{Discrim}}$, the discriminative module of NTPP: The network traffic dynamics not only depend on the collective packet transmission processes of several hosts, but also the contention between these hosts vying with each other for the network bandwidth. Such a contention manifests relative variation in *bandwidth rankings* of the contending hosts over time. By the term *bandwidth ranking*, we indicate the relative rankings of the contending hosts in terms of their total bandwidth share of the available capacity over time. To model these ranking dynamics, we first split the interval $[0, T)$, into L sets of small, equal and disjoint sub-intervals $[0, T_s), [T_s, 2T_s), \ldots, [(L-1)T_s, T)$, where $T_s = T/L$. Then, we define

$$z^i_{\boldsymbol{H},\boldsymbol{H}'} = \mathbb{I}(N_{\boldsymbol{H}}[iT_s, (i+1)T_s) > N_{\boldsymbol{H}'}[iT_s, (i+1)T_s)),$$
$$\boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'} = \int_{iT_s}^{(i+1)T_s} (\lambda_{\boldsymbol{H}}(t) - \lambda_{\boldsymbol{H}'}(t))dt \tag{6}$$

Here $\mathbb{I}[x]$ is the indicator function which is equal to 1 (0) if x is true (false). Furthermore, $z^i_{\boldsymbol{H},\boldsymbol{H}'} = 1$ (0) indicates that the *observed* bandwidth of $\boldsymbol{H}$ is greater (less) than that of $\boldsymbol{H}'$ during $[iT_s, (i+1)T_s)$. On the other hand, $\boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'}$ *estimates* the difference in bandwidth between $\boldsymbol{H}$ and $\boldsymbol{H}'$. Now, for each interval $[iT_s, (i+1)T_s)$, we employ several ranking loss functions defined in Table I to capture the ranking variations of hosts (i.e. variations of $z^i_{\boldsymbol{H},\boldsymbol{H}'}$ across $0 \leq i < L$). These loss functions are often used in learning to rank in the context of information retrieval [17]. They encode different measures for ranked list of items, by comparing pairwise order between each pair of items. In our application, we use these measures to model the relative order among the competing hosts, induced by their contention process.

Unbiased ranking model: A simple unbiased ranking function may be derived following a large margin approach. Suppose we are given two hosts $\boldsymbol{H}_1, \boldsymbol{H}_2 \in \mathbb{H}$ and time instances t_s and t_f ($t_s < t_f$),

$$N_{\boldsymbol{H}_1}[t_s, t_f) \geq N_{\boldsymbol{H}_2}[t_s, t_f) \implies \int_{t_s}^{t_f} \lambda_{\boldsymbol{H}_1}(t)dt \geq \int_{t_s}^{t_f} \lambda_{\boldsymbol{H}_2}(t)dt$$

Here, $N_{\boldsymbol{H}}[t_s, t_f)$ indicates the number of transmission events of host $\boldsymbol{H}$ occurring in the interval $[t_s, t_f)$, and is approximated using $\int_{t_s}^{t_f} \lambda_{\boldsymbol{H}_1}(t)dt$ (Eq. 2). In words, if the number of transmitted events of $\boldsymbol{H}_1$ is more than $\boldsymbol{H}_2$, so are their estimates $\int_{t_s}^{t_f} \lambda_*(t)dt$. There are several ways to capture such scenarios. The easiest way is to plug them as constraints, while estimating the parameters using the likelihood function. Another plausible direction is to follow the probabilistic approach that we adopt in our proposed model. We wish to minimize the number of instances, where $z^i_{\boldsymbol{H},\boldsymbol{H}'}$ and $\boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'}$ are not in agreement. That is, we maximize the hard ranking loss given in Table I which, however, is not differentiable. By approximating it using sigmoids we define the discriminative loss function as a soft ranking loss, $\boldsymbol{D}_{\text{Unbiased}}(i)$ (see Table I). Here, $\boldsymbol{\sigma}(w; x) = 1/(1 + \exp(-w.x))$ is the sigmoid function which approximates the indicator function $\mathbb{I}(x > 0)$, and the weight w controls the slope of the sigmoid. As $w \to \infty$, $\sigma(w; x) \to \mathbb{I}(x > 0)$.

An unbiased ranking model is useful when the variation of bandwidth across $\mathbb{H}$ is uniform, i.e. when for a randomly chosen host $\boldsymbol{H}$, the numbers of hosts having a high bandwidth-share remain nearly the same with those having a low bandwidth-share (eg. ISP backbone [11]). For skewed traffic dynamics where there are only a few heavily contending hosts (ex. iSCSI based storage system [13]), we employ more complex ranking losses, e.g. precision, recall, area under ROC curve (AUC), etc. that are more reliable measures in ranking imbalanced data.

Precision: Precision measures how accurately the estimated rank of a host indicates its actual relative position against other hosts in terms of the number of transmitted packets. Given a host $\boldsymbol{H}$, Precision($\boldsymbol{H}, i$) is defined as the probability that it actually sends more packets than a randomly selected host $\boldsymbol{H}'$, given that $\boldsymbol{H}$ is estimated to send more packets than $\boldsymbol{H}'$, during the interval $[iT_s, (i+1)T_s)$. Formally, the hard ranking loss in Table I is equal to 1-Precision($\boldsymbol{H}, i$), which again is a non-smooth ranking function that can be approximated using sigmoid, yielding the corresponding soft ranking loss $\boldsymbol{D}_{\text{Precision}}(\boldsymbol{H}, i)$.

Recall: Given a host $\boldsymbol{H}$, Recall($\boldsymbol{H}, i$) is defined as the probability that it is estimated to send more packets than a randomly selected host $\boldsymbol{H}'$, given that it actually sends more packets than $\boldsymbol{H}'$, during the time interval $[iT_s, (i+1)T_s)$. Formally, the hard ranking loss in Table I is equal to 1-Recall($\boldsymbol{H}, i$), which is turned into a differentiable soft ranking loss $\boldsymbol{D}_{\text{Recall}}(\boldsymbol{H}, i)$.

AUC: Before going to AUC, we first connect the idea of ROC (receiver operating characteristics) in the context of network traffic dynamics. Suppose, we obtain a ranked list of $\boldsymbol{H} \in \mathbb{H}$ according to the value of $\int_{iT_s}^{(i+1)T_s} \lambda_{\boldsymbol{H}}(t)dt$. Then at each position $k \in \{1, ..., |\mathbb{H}|\}$, we compute the true

TABLE I: Ranking losses for the discriminative model for the interval $[iT_s, (i+1)T_s)$. For Measure $\in$ {Precision, Recall, AUC}, the hard ranking loss is defined as 1 - (Measure). The soft ranking loss is a smooth approximation of the same.

Measure	**Ranking loss (Hard)**	**Ranking loss (Soft)** ($\boldsymbol{D}_{\text{Measure}}$)
Unbiased	$\prod_{\boldsymbol{H},\boldsymbol{H}' \in \mathbb{H}} (1 - z^i_{\boldsymbol{H},\boldsymbol{H}'} \mathbb{I}[\zeta^i_{\boldsymbol{H},\boldsymbol{H}'} < 0])$	$\prod_{\boldsymbol{H},\boldsymbol{H}' \in \mathbb{H}} (1 - z^i_{\boldsymbol{H},\boldsymbol{H}'} \boldsymbol{\sigma}[\boldsymbol{W}; -\boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'}])$
Precision	$1 - \frac{\sum_{\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'} \mathbb{I}[\zeta^i_{\boldsymbol{H},\boldsymbol{H}'} > 0]}{\sum_{\boldsymbol{H}'} \mathbb{I}[\zeta^i_{\boldsymbol{H},\boldsymbol{H}'} > 0]}$	$1 - \frac{\sum_{\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'} \boldsymbol{\sigma}(\boldsymbol{W}; \boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'})}{\sum_{\boldsymbol{H}'} \boldsymbol{\sigma}(\boldsymbol{W}; \boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'})}$
Recall	$1 - \frac{\sum_{\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'} \mathbb{I}[\zeta^i_{\boldsymbol{H},\boldsymbol{H}'} > 0]}{\sum_{\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'}}$	$1 - \frac{\sum_{\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'} \boldsymbol{\sigma}(\boldsymbol{W}; \boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'})}{\sum_{\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'}}$
AUC	$\frac{\sum_{\boldsymbol{H},\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'} \mathbb{I}[\zeta^i_{\boldsymbol{H},\boldsymbol{H}'} < 0]}{\lvert\mathbb{H}\rvert^2}$	$\frac{\sum_{\boldsymbol{H},\boldsymbol{H}'} z^i_{\boldsymbol{H},\boldsymbol{H}'} \boldsymbol{\sigma}(\boldsymbol{W}; -\boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'})}{\lvert\mathbb{H}\rvert^2}$

positive rate $\text{TPR}_k := \mathbb{P}_k(\boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'} > 0, z^i_{\boldsymbol{H},\boldsymbol{H}'}, > 0)$ for two randomly selected hosts $\boldsymbol{H}$ and $\boldsymbol{H}'$ within rank k, as well as the corresponding false positive rate $\text{FPR}_k := \mathbb{P}_k(\boldsymbol{\zeta}^i_{\boldsymbol{H},\boldsymbol{H}'} < 0, z^i_{\boldsymbol{H},\boldsymbol{H}'} > 0)$. The variation of TPR_k (Y axis) with FPR_k (X axis) provides the ROC curve in our context. The area under the ROC curve, AUC(i), offers an important measure in the context of any ranking paradigm. We derive the hard ranking loss (see Table I) as 1-AUC(i) using a well known formula for AUC [30], and a corresponding soft ranking loss $\boldsymbol{D}_{\text{AUC}}(i)$.

Precision of the process is important when the interval $[iT_s, (i+1)T_s)$ is very short, since then the model makes predictions very frequently (so we can afford to miss some ground-truth provided we make accurate predictions). This loss function should therefore be used when the nature of the traffic is very volatile and changes rapidly with time. Recall is important when the interval $[iT_s, (i+1)T_s)$ is long, since then predictions are few and far between (so we cannot miss the ground truth even at the cost of making additional errors). This loss function should be used for traffic showing well-defined and consistent patterns. AUC, on the other hand, is intermediate between the two and should be used when the traffic variability is moderate.

C. Inference of the Generative Process

Consider a set of N hosts $\mathbb{H} = \{\boldsymbol{H}_l | 1 \leq l \leq N\}$, associated with a collection of transmission events $\mathcal{H}_{\boldsymbol{H}_l}(T) = \{t_i\}$ for each host $\boldsymbol{H}_l$ during a time period $[0, T)$. Using these timestamps, we attempt to infer the underlying generative process, as well as the parameters of the discriminative ranking model. To learn the parameters, we maximize the log-likelihood for the recorded events $\cup_{\boldsymbol{H} \in \mathbb{H}} \mathcal{H}_{\boldsymbol{H}}(T)$ combined with the ranking losses. Here, the log-likelihood function is

$$\log[L(\boldsymbol{a}_{\bullet,\bullet}, \boldsymbol{b}_{\bullet,\bullet}, \boldsymbol{w}, \boldsymbol{\xi}, \boldsymbol{l}_{\bullet}, \boldsymbol{\beta}_{\bullet}, \boldsymbol{\gamma}_{\bullet}, \boldsymbol{\omega}_{\bullet}, b, c)] = \sum_{\boldsymbol{H} \in \mathbb{H}} \sum_{t_i \in \mathcal{H}_{\boldsymbol{H}}(T)} \log \lambda_{\boldsymbol{H}}(t_i) - \sum_{\boldsymbol{H} \in \mathbb{H}} \int_0^T \lambda_{\boldsymbol{H}}(t) dt. \quad (7)$$

All the variables in the arguments of $L(.)$ are the parameters of the neural networks in the generative model. To incorporate the effect of competing hosts, we again split the interval $[0, T)$, into L sets of small, equal and disjoint subintervals $[0, T_s), [T_s, 2T_s), \ldots, [(L-1)T_s, T)$, where $T_s = T/L$, where for each interval, we have a corresponding discriminative loss $\boldsymbol{D}_{\text{Measure}}(\boldsymbol{H}, i)$, for Measure $\in$ {Precision, Recall}, or $\boldsymbol{D}_{\text{Measure}}(i)$ where Measure $\in$ {Unbiased, AUC}. Finally, we compute the parameters by minimizing the generative loss (negative log-likelihood) simultaneously with the discriminative loss in the following way:

$$\min_{\boldsymbol{W},\boldsymbol{G}} - \log L(\cup_{\boldsymbol{H} \in \mathbb{H}} | \boldsymbol{G}) + \sum_{i=1}^{L} \sum_{\boldsymbol{H} \in \mathbb{H}} \log(\boldsymbol{D}_{\text{Measure}}(\boldsymbol{H}, i))$$

Here $\boldsymbol{G}$ is given by the neural parameters of the generative model. In the next section, we analyze the effectiveness of this model over various real-world datasets.

IV. Experiments

In this section, we first describe the experimental setup illustrating the datasets, the evaluation metrics, and the baseline methods. Then, we provide a detailed comparative analysis of NTPP against these baselines.

A. Experimental setup

We evaluate our model on seven datasets. Four of these are collected from Cyber-defense exercises (CDX), as listed by Netresec [31]. The remaining three are *real-world* datasets, one from website access logs, one from datacenter traffic, and one from P2P traffic. The details follow.

MACCDC[1]: It consists of packet captures (pcaps) from National CyberWatch Mid-Atlantic Collegiate Cyber Defense competition held in 2010 (17 pcap files, $\sim$5.4 GB), 2011 (15 pcap files, $\sim$14.2 GB) and 2012 (27 pcap files, $\sim$10.1 GB). This is a server traffic dataset, where attacks have been injected from external machines.

ISTS[2]: It consists of pcaps from the Information Security Talent Search (ISTS) competition which is an annual three day cyber attack/defend competition hosted at the Rochester Institute of Technology by SPARSA. We took a sample of the data for two days of the competition held on March 7, 2015 (10 pcap files, 1.36 GB) and March 8, 2015 (10 pcap files, 1.52 GB). This is a local network dataset, where students have accessed various services involving system administration, remote service access, server based programming etc.

WRCCDC[3]: It consists of pcaps from the Western Regional Collegiate Cyber Defense Competition (over 1TB of pcaps) held in 2017 and 2018. We took a sample of 10 pcaps from 2017 (4.32 GB) and 10 pacps from 2018 (4.77 GB). This is a local network dataset with web and mail servers, where attacks have been injected from the external machines.

[1] https://www.netresec.com/?page=MACCDC (last accessed: January 18, 2019)

[2] https://www.netresec.com/?page=ISTS (last accessed: January 18, 2019)

[3] https://archive.wrccdc.org/pcaps/ (last accessed: January 18, 2019)

ISACDC[4]**:** This dataset consists of the packet captures from the "2009 Inter-Service Academy Cyber Defense Competition" held on April 21-24, 2009 served by Information Technology Operations Center (ITOC), United States Military Academy, which amount to ∼12GB of captured data, along with Snort IDS logs, DNS logs, Web Server logs, and a Log Server Aggregate log.

WorldCup98[5]**:** This dataset consists of binary log files collected for 92 days (between April 30, 1998 and July 26, 1998) of the 1998 FIFA World Cup website (www.france98.com) by Martin Arlitt. During this period of time the site received $1,352,804,107$ requests. There are altogether 249 binary log files, of which the first four are empty, with the total amount of traffic being 4991 GB.

UNIDC[6]**:** This dataset consists of packet traces from two university datacenters. There are 22 pcaps totalling 2.08 GB from the first datacenter and 9 pcaps totalling 8.47 GB from the second.

snu/bittorrent [32]: This is a traceset of packet captures of BitTorrent traffic from Korea Telecom's mobile WiMAX network in Seoul from March 2010, with payload cut-off and with headers anonymized, available from CRAWDAD. The dataset has 5 traces with 24 pcaps, totalling 0.51 GB.

NTPP and Baselines: Our proposal, which contains a discriminative model apart from a generator, operates over a variety of ranking functions, i.e. unbiased ranking losses, precision, recall, and AUC. We call the corresponding derivatives of our model NTPP_{ub}(unbiased ranking loss), NTPP_{pr}(precision loss), NTPP_{re}(recall loss), and NTPP_{auc}(AUC loss), respectively. In the result section, we report the performance of the best among the variants of NTPP. We compare NTPP with three baseline models for network traffic prediction, viz.: (i) Wavelet Transform with Artificial Neural Networks (WT+ANN) [15], (ii) Sequential Tensor Completion (STC) [9], and (iii) Autoregressive Integrated Moving Average (ARIMA) [21] Other baselines, which are based on temporal point processes, and have been used in other fields such as social networks for predicting bursty events, are (iv) Wasserstein GAN Temporal Point Process (WGANTPP) [33], (v) Sister-Tweet Reinforcement Model (STRM) [27], and (vi) Hawkes process [34], To evaluate the performance of the generative model in absence of the discriminator, we consider as a baseline (vii) NTPP_{Gen}, which is the generative model of NTPP trained only to maximize the log-likelihood without any feedback. Since (viii) RMTPP [14] is very closely related to NTPP_{Gen}, we consider it separately so that we may use it as a baseline specifically to evaluate the performance of NTPP_{Gen}.

Experiment methodology: In each dataset, we use the first 80% of the total volume of events to train the model which, thereafter, is used to test the next 20% events. To predict the dynamics in future, we use Ogata's thinning algorithm [35] to simulate the learned model. As the first predictive task, we predict the future traffic volume of hosts by calculating the total traffic volume in the simulated trace. We then extend these experiments for predicting the rank order of the hosts, including prediction of jumps in ranking on the CDX datasets where modeling individual hosts is especially significant to identify malicious hosts. Finally we also perform traffic spike detection on the real-world (P2P, Web, Data-center) datasets, where modeling the overall traffic is a greater concern. Thereafter, we attempt to predict the next transmission events of hosts through the method of [36]. This method has been developed for point processes, which we extend to the networking field. Prediction of the next transmission event is useful to identify periods of inactivity or traffic bursts from hosts. This has direct implications for downstream caching applications, as we show later.

[4]https://www.westpoint.edu/crc/SitePages/DataSets.aspx (last accessed: January 18, 2019)

[5]http://ita.ee.lbl.gov/html/contrib/WorldCup.html (last accessed: January 18, 2019)

[6]http://pages.cs.wisc.edu/ tbenson/IMC10˙Data.html (last accessed: January 18, 2019)

TABLE II: $\text{MAPE}_{\text{count}}$ (%) of the best-performing variant of NTPP and baseline algorithms; cells with light green (yellow) color indicate the best (second best) predictor. Numbers in bracket indicate the percentage improvement from the second best predictor.

	$\text{MAPE}_{\text{count}}$ (%)						
Datasets	**NTPP**	**WGANTPP**	**STRM**	**WT+ANN**	**Hawkes**	**STC**	**ARIMA**
MACCDC	**12.78** (17.71%)	**15.53**	16.23	21.54	16.76	21.72	24.74
ISTS	**5.03** (16.72%)	**6.04**	6.43	13.46	18.68	13.16	22.48
WRCCDC	**4.54** (13.52%)	5.67	**5.25**	9.05	14.84	18.42	26.68
ISACDC	**5.18** (6.67%)	5.78	**5.55**	8.65	14.67	19.42	18.54
WorldCup98	**5.12** (10.80%)	**5.74**	6.22	8.29	15.67	20.55	23.77
UNIDC	**5.15** (7.04%)	6.18	**5.54**	8.89	14.66	18.56	22.18
snu/bittorrent	**4.86** (4.52%)	5.45	**5.09**	8.68	9.64	8.85	21.46

B. Experiment 1: Forecasting host traffic

In this experiment, the predictive model is used to forecast host traffic by computing the expected number of host transmission events in the test set. We measure *Mean Absolute Percentage Error for event count (*MAPE_{count}*)* [27], which represents the normalized mean absolute deviation between the observed and the predicted cumulative packet counts for a host up to the current time instant. We note that this is a continuous-time metric and does not require the time to be sliced into discrete intervals.

Table II dissects a comparative analysis of different methods in terms of $\text{MAPE}_{\text{count}}$ across all the datasets. It shows that NTPP performs best against all its competitors, by achieving the lowest $\text{MAPE}_{\text{count}}$. We observe that the performances of WT+ANN, Hawkes, STC and ARIMA are substantially poor. This is because of the characteristics of the datasets under consideration. WT+ANN, STC and ARIMA are not suited for bursty traffic. Hawkes, on the other hand, is a very constrained model and cannot model the complexity of the process. NTPP and STRM, on the other hand, are sufficiently elaborate models which can handle irregular traffic and so perform well.

C. Experiment 2: Rank prediction of contending hosts

In this experiment, we attempt to predict the dominance rankings of contending hosts in the test set. In each test interval, we obtain a ranked list of the network hosts, based on the traffic predicted using our proposed models as well as the

TABLE III: SRCC of the best-performing variant of NTPP and baseline algorithms; cells with light green (yellow) color indicate the best (second best) predictor. Numbers in bracket indicate the percentage improvement from the second best predictor.

	SRCC						
Datasets	**NTPP**	**WGANTPP**	**STRM**	**WT+ANN**	**Hawkes**	**STC**	**ARIMA**
MACCDC	**0.91** (7.05%)	**0.85**	0.84	0.72	0.51	0.62	0.37
ISTS	**0.93** (4.49%)	0.88	**0.89**	0.54	0.46	0.51	0.36
WRCCDC	**0.89** (4.70%)	0.81	**0.85**	0.73	0.51	0.68	0.38
ISACDC	**0.88** (4.76%)	0.78	**0.84**	0.69	0.49	0.40	0.35
WorldCup98	**0.93** (3.33%)	**0.90**	0.89	0.71	0.78	0.40	0.32
UNIDC	**0.92** (2.22%)	0.88	**0.90**	0.79	0.84	0.42	0.35
snu/bittorrent	**0.89** (5.95%)	**0.84**	0.81	0.72	0.79	0.49	0.31

baselines. We also derive the ground-truth of these ranked lists and compare with the predicted rank order using *Spearman's Rank Correlation Coefficient* (SRCC) [37] which captures the ranking dynamics in each time-interval.

Table III depicts a comparative sketch of the predictive powers of different models in terms of SRCC. It reflects the ability of the algorithms to probe variations in host rankings during the contention process. We observe that NTPP performs substantially better than all the baselines across all the datasets. Similar to $\text{MAPE}_{\text{count}}$, in this case too, both WGANTPP and STRM perform reasonably well and second NTPP. As in Experiment 1, STC, WT+ANN and ARIMA do not perform well here, since they are not suitable for such irregular traffic. Hawkes does not consider the contention mechanism between hosts and so fails to perform well.

D. Experiment 3: Jump detection

In the CDX datasets, we measure the ability of our models to predict the instances where a particular host suddenly gains dominance (which may happen when there is a spike in the traffic generated by the host, especially if it tries to flood the network with a DDoS attack) or suddenly loses dominance (which may happen when the host shuts down due to effect of cyber-attacks). Note that in the first instance, the host is a likely attacker, while in the second case, it is a likely victim.

Let a host $\boldsymbol{H} \in \mathbb{H}$ have a rank $r_{\boldsymbol{H},[t_i,t_{i+1})}$ in the time-interval $[t_i, t_{i+1})$. Then, if $|r_{\boldsymbol{H},[t_i,t_{i+1})} - r_{\boldsymbol{H},[t_{i-1},t_i)}| \geq |\mathbb{H}|/2$, it is considered a *jump*. We compare the set of predicted jumps with the set of actual jumps and use average recall and average precision to measure our prediction performance: Recall measures the proportion of real jumps which are correctly identified by an algorithm, while precision measures the fraction of cases where the jump predicted by an algorithm is actually observed in the real data.

The first 4 rows of Table VI report the average precision and recall of jump detection for NTPP and baselines across the CDX datasets on the 20% held-out test set. We note that, STRM outperforms the rest of the baselines in terms of both precision and recall, making it the most competitive baseline, followed by WGANTPP. This is because STRM explicitly models the inter-host contention, whereas WGANTPP does not. As expected, WT+ANN, STC and ARIMA do not perform well here.

The stellar performance of NTPP in jump detection is expected since the underlying discriminator is especially geared towards capturing inter-host contention. Since the loss functions used by all variants of our approach deal with pairwise

TABLE IV: $\text{MAPE}_{\text{count}}$ of several variants of NTPP along with RMTPP (used as a baseline for NTPP_{Gen}); cells with light green (yellow) color indicate the best (second best) predictor.

	$\text{MAPE}_{\text{count}}$ (%)					
Datasets	**NTPP_{ub}**	**NTPP_{pr}**	**NTPP_{re}**	**NTPP_{auc}**	**NTPP_{Gen}**	**RMTPP**
MACCDC	13.15	**12.78**	13.38	**12.17**	13.56	13.98
ISTS	5.09	**5.03**	**5.05**	5.14	5.18	5.26
WRCCDC	**4.67**	5.26	**4.54**	5.24	5.58	5.96
ISACDC	6.19	**5.58**	**5.18**	6.22	6.26	6.48
WorldCup98	5.24	**5.12**	5.20	**5.14**	5.26	5.44
UNIDC	6.17	6.05	**5.15**	**5.98**	6.22	6.45
snu/bittorrent	4.98	**4.86**	5.04	**4.90**	5.27	5.56

TABLE V: SRCC of several variants of NTPP along with RMTPP (used as a baseline for NTPP_{Gen}); cells with light green (yellow) color indicate the best (second best) predictor.

	SRCC					
Datasets	**NTPP_{ub}**	**NTPP_{pr}**	**NTPP_{re}**	**NTPP_{auc}**	**NTPP_{Gen}**	**RMTPP**
MACCDC	0.84	0.89	**0.90**	**0.91**	0.71	0.68
ISTS	**0.89**	**0.93**	0.81	0.84	0.76	0.70
WRCCDC	0.84	0.86	**0.89**	**0.88**	0.84	0.79
ISACDC	0.81	0.84	**0.86**	**0.88**	0.70	0.63
WorldCup98	0.86	0.88	**0.89**	**0.93**	0.81	0.75
UNIDC	0.78	**0.92**	**0.88**	0.83	0.79	0.72
snu/bittorrent	0.84	**0.88**	0.86	**0.89**	0.82	0.71

interactions between hosts, they can more accurately predict when a host is going to be significantly promoted or demoted in rank relative to the other hosts.

E. Experiment 4: Traffic spike detection

In the real datasets, we measured the ability of our models to detect sudden spikes in the combined traffic from all the hosts on the network. Note that this is a less fine-grained analysis than Experiment 3, but is suitable for real-world traffic in general. We say a spike has occurred when the net traffic volume in an interval exceeds that in the previous interval by more than 50%. The metrics used here are average recall and average precision (for spike detection). The interpretations are analogous to Experiment 3.

The last three rows of Table VI report the average precision and recall of spike detection for NTPP and baselines on the WorldCup98, UNIDC and snu/bittorrent datasets. The trends in performance observed here are the same as those observed for jump detection. We note that the existing traffic prediction approaches, viz. WT+ANN, STC and ARIMA fail to detect such traffic spikes effectively, due to the limiting assumption of self-similarity.

F. Experiment 5: Next-event prediction

As in [36], we can predict the time, $\hat{t}_{\boldsymbol{H},i+1}$, of the next transmission event of a host, given its history upto the last event at time $t_{\boldsymbol{H},i}$ using the conditional intensity. In case of STC, WT+ANN and ARIMA, this is done by learning the pattern of inter-arrival times instead of traffic volume. We measure *Mean Absolute Percentage Error for event time (*MAPE_{time}*)* (analogous to $\text{MAPE}_{\text{count}}$) which represents the normalized mean absolute deviation between the observed and the predicted packet arrival times, $t_{\boldsymbol{H},i}$ for a host $\boldsymbol{H}$ up to the current time instant. Like $\text{MAPE}_{\text{count}}$, this too is a continuous-time metric and does not require time discretization into intervals.

Table VIII reports the results using the metric $\text{MAPE}_{\text{time}}$ a comparison between the best-performing variant and the baselines. Unlike other experiments, here WGANTPP consistently

TABLE VI: Average precision and recall in jump (in datasets marked with †) / spike (in datasets marked with *) detection for NTPP and baselines. The cells with light green (yellow) color indicate the best (second best) predictor. Numbers in bracket indicate the percentage improvement from the second best predictor.

	Avg. Precision							Avg. Recall						
Datasets	**NTPP**	**WGANTPP**	**STRM**	**WT+ANN**	**Hawkes**	**STC**	**ARIMA**	**NTPP**	**WGANTPP**	**STRM**	**WT+ANN**	**Hawkes**	**STC**	**ARIMA**
MACCDC †	**0.82** (15.49%)	0.62	**0.71**	0.51	0.33	0.29	0.32	**0.84** (16.67%)	0.63	**0.72**	0.29	0.32	0.31	0.34
ISTS †	**0.85** (25.00%)	0.56	**0.68**	0.40	0.26	0.28	0.27	**0.83** (18.57%)	0.48	**0.70**	0.24	0.28	0.31	0.32
WRCCDC †	**0.84** (25.37%)	0.61	**0.67**	0.32	0.23	0.38	0.40	**0.86** (26.47%)	0.60	**0.68**	0.30	0.27	0.29	0.34
ISACDC †	**0.82** (17.14%)	0.59	**0.70**	0.38	0.28	0.39	0.34	**0.89** (23.61%)	0.64	**0.72**	0.38	0.26	0.44	0.32
WorldCup98*	**0.83** (27.7%)	0.53	**0.65**	0.37	0.23	0.25	0.24	**0.80** (19.4%)	0.45	**0.67**	0.21	0.25	0.28	0.29
UNIDC*	**0.81** (26.6%)	0.58	**0.64**	0.29	0.20	0.35	0.37	**0.83** (27.7%)	0.57	**0.65**	0.27	0.24	0.26	0.31
snu/bittorrent*	**0.79** (17.9%)	0.56	**0.67**	0.35	0.25	0.36	0.31	**0.86** (24.6%)	0.61	**0.69**	0.35	0.23	0.41	0.29

TABLE VII: Average precision and recall in jump (in datasets marked with †) / spike (in datasets marked with *) detection for all variants of NTPP. Bold (italics) indicates best (second best) predictor.

	Avg. Precision (Avg. Recall)			
Datasets	**$NTPP_{ub}$**	**$NTPP_{pr}$**	**$NTPP_{re}$**	**$NTPP_{auc}$**
MACCDC †	0.73 (0.75)	**0.82** (0.78)	0.70 (**0.84**)	*0.79 (0.78)*
ISTS †	0.69 (0.65)	**0.85** (0.70)	0.71 (**0.83**)	*0.80 (0.79)*
WRCCDC †	0.71 (0.73)	**0.84** (0.72)	0.68 (**0.86**)	*0.79 (0.78)*
ISACDC †	0.70 (0.69)	**0.82** (0.71)	0.76 (**0.89**)	*0.79 (0.79)*
WorldCup98*	0.66 (0.62)	**0.82** (0.67)	0.68 (**0.80**)	*0.77 (0.76)*
UNIDC*	0.68 (0.70)	**0.81** (0.69)	0.65 (**0.83**)	*0.76 (0.75)*
snu/bittorrent*	0.67 (0.66)	**0.79** (0.68)	0.73 (**0.86**)	*0.76 (0.76)*

TABLE VIII: $MAPE_{time}$ (%) of the best-performing variant of NTPP and the baselines; cells with light green (yellow) color indicate the best (second best) predictor. Numbers in bracket indicate the percentage improvement from the second best predictor.

	$MAPE_{time}$ (%)						
Datasets	**NTPP**	**WGANTPP**	**STRM**	**WT-ANN**	**Hawkes**	**STC**	**ARIMA**
MACCDC	**5.76** (6.34%)	**6.15**	6.78	6.48	6.68	6.23	7.76
ISTS	**5.03** (13.57%)	**5.82**	6.49	6.54	6.34	6.28	7.13
WRCCDC	**4.64** (18.16%)	**5.67**	6.06	6.20	6.24	6.35	6.48
ISACDC	**5.16** (16.50%)	**6.18**	7.19	7.46	8.22	8.21	8.26
WorldCup98	**5.12** (10.80%)	**5.74**	6.15	6.19	6.24	5.95	6.87
UNIDC	**4.95** (18.58%)	**6.08**	6.16	6.21	7.00	6.24	7.14
snu/bittorrent	**4.86** (17.49%)	**5.89**	6.51	6.58	6.96	6.74	7.21

performs the second-best due to its strong predictive power, while STRM shows poor performance, falling behind STC in some cases. This is because this experiment involves extremely short-range predictions which STRM cannot model due to the lack of sufficient contention data. However, STC is good for such applications since it is primarily a data recovery model.

G. Comparison across the variants of NTPP

To have a better understanding about our model, we perform a comparative study among the performances of several variants of NTPP with different discriminative losses. We also consider the performance of $NTPP_{Gen}$, the generative-only version of NTPP in the context of RMTPP which serves a baseline for $NTPP_{Gen}$. This is illustrated in the Tables IV, V and VII .

From Tables IV and V, we observe that precision and recall turn out to be the best discriminative measures, when the performance is compared in terms of $MAPE_{count}$ as well as $MAPE_{time}$. However, in case of SRCC, AUC performs significantly better against all other ranking measures. This is because, AUC is a more powerful ranking measure than both precision and recall, and as a result, it facilitates NTPP to capture the ranking dynamics more effectively than others. We also observe that $NTPP_{ub}$ usually fares poorly due to its inability to capture the skewed ranking distribution present in most of the cases. $NTPP_{Gen}$ does not capture the host contention process at all, which severely affects its performance in terms of $MAPE_{count}$, $MAPE_{time}$ and SRCC. However, we can see that it consistently performs better than RMTPP in all cases, thereby justifying its incorporation as a component of NTPP in preference to using RMTPP [14] itself as a generator. From Table VII, we observe that $NTPP_{pr}$ reports the highest precision in all datasets, while, $NTPP_{re}$ reports the highest recall across all datasets. It is interesting to note that $NTPP_{auc}$ always reports the second-highest precision and recall.

H. Downstream application: Caching performance

In order to examine the impact of our model on a caching application, we simulate a network consisting of 4 hosts and a web server equipped with a 100MB LRU cache using the widely-known network simulator NS-3. The cache is implemented in two ways: (i) the traditional *host-agnostic* cache which does not distinguish between different hosts and (ii) a *preemptive host-aware* cache which recognizes the identity of hosts and splits itself into 4 virtual caches, one for each host, with different sizes (which sum to the total cache size). These cache sizes are determined dynamically based on two cache size update schemes – (i) **discrete time update** and (ii) **continuous time update**. In the **discrete time update** scheme, the cache sizes are updated at fixed intervals of 10 minutes. The amount of memory allocated to each host at the end of every interval is decided based on the ratio of their forecast traffic volumes for the next interval, based on their history up to that point. In the **continuous time update** scheme, the cache sizes are updated whenever a packet arrives on the network, and the allocation is done in the inverse proportion of the predicted next-packet arrival times for each host (based on the method used in Experiment 5). The intuition behind this is that, if a packet is expected sooner from a host, it should be given more cache and vice versa. In this way, we collect cache performance statistics for a span of 5 hours and note the net cache miss rate in both cases. In order to simulate real web traffic, we arbitrarily selected 4 hosts on a given day from the WorldCup98 dataset with enough traffic and replicated their HTTP request patterns on our network with the same timestamps with simulated target pages.

We observed that, at the end of 5 hours, the host-agnostic cache logged a miss rate of 22.57%, while the host-aware cache schemes logged a miss rate of 12% - 15% depending on the NTPP variant used. In general, the continuous time update scheme proved to be superior, but it is also computationally more expensive. The results are shown in Table IX. We see that $NTPP_{re}$ performs best for discrete time update, while $NTPP_{pr}$ performs best for continuous time update, with $NTPP_{auc}$ performing second-best in both cases. This is because, in the continuous time case, precision is more important since the

TABLE IX: Miss rates (%) obtained by using a pre-emptive host-aware cache aided by NTPP variants over the host-agnostic cache. The miss rate obtained by the host-agnostic cache is 22.57% .The cells with light green (yellow) color indicate the best (second best) predictor.

Cache update scheme	NTPP_{ub}	NTPP_{pr}	NTPP_{re}	NTPP_{auc}
Discrete time update	14.89	14.38	**14.21**	**14.35**
Continuous time update	12.67	**12.15**	12.42	**12.27**

prediction is very short-range and is done very frequently (so high coverage is not essential). Whereas for the discrete time case, recall is more important since we are dealing with events in bulk and the prediction is done only occasionally (so we cannot afford to have low coverage).

V. Conclusion

In this paper, we propose NTPP, a novel probabilistic modeling framework for predicting network traffic dynamics. Our model is specially suited to traffic exhibiting highly anomalous patterns, such as attack traffic, datacenter microbursts etc. We show in our experiments that existing approaches to traffic prediction fare poorly in such scenarios, since they rely on the assumption of self-similarity of network traffic, whereas our model fares much better. We also demonstrate an example application of our model to enhance the performance of a downstream caching application.

References

[1] D. Shan and F. Ren, "Improving ECN marking scheme with micro-burst traffic in data center networks," in *IEEE INFOCOM*, 2017, pp. 1–9.

[2] J. Zhang, F. R. Yu, S. Wang, T. Huang, Z. Liu, and Y. Liu, "Load balancing in data center networks: A survey," *IEEE Communications Surveys & Tutorials*, 2018.

[3] A. Fumo, M. Fiore, and R. Stanica, "Joint spatial and temporal classification of mobile traffic demands," in *IEEE INFOCOM*, 2017, pp. 1–9.

[4] T. Zhao, Q. Liu, and C. W. Chen, "QoE in video transmission: A user experience-driven strategy," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 285–302, 2017.

[5] B. Hou, F. Chen, Z. Ou, R. Wang, and M. Mesnier, "Understanding I/O performance behaviors of cloud storage from a client's perspective," *ACM trans. on Storage*, vol. 13, no. 2, p. 16, 2017.

[6] R. de Toledo Caropreso, R. A. S. Fernandes, D. P. M. Osorio, and I. N. da Silva, "An open source framework for smart meters: Data communication and security traffic analysis," *IEEE trans. on Industrial Electronics*, 2018.

[7] C. Kolias, G. Kambourakis, A. Stavrou, and J. Voas, "Ddos in the iot: Mirai and other botnets," *Computer*, vol. 50, no. 7, pp. 80–84, 2017.

[8] T. Benson, A. Anand, A. Akella, and M. Zhang, "MicroTE: Fine grained traffic engineering for data centers," in *ACM CoNEXT*, 2011, p. 8.

[9] K. Xie, L. Wang, X. Wang, G. Xie, J. Wen, and G. Zhang, "Accurate recovery of internet traffic data: A tensor completion approach," in *IEEE INFOCOM*, 2016, pp. 1–9.

[10] Q. Zhang, V. Liu, H. Zeng, and A. Krishnamurthy, "High-resolution measurement of data center microbursts," in *Proceedings of the ACM IMC*, 2017, pp. 78–85.

[11] H. Abrahamsson, B. Ahlgren, P. Lindvall, J. Nieminen, and P. Tholin, "Traffic characteristics on 1gbit/s access aggregation links," in *IEEE ICC*, 2017, pp. 1–7.

[12] F. Xu, Y. Li, H. Wang, P. Zhang, and D. Jin, "Understanding mobile traffic patterns of large scale cellular towers in urban environment," *IEEE/ACM trans. on netw.*, vol. 25, no. 2, pp. 1147–1161, 2017.

[14] N. Du, H. Dai, R. Trivedi, U. Upadhyay, M. Gomez-Rodriguez, and L. Song, "Recurrent marked temporal point processes: Embedding event history to vector," in *Proceedings of the 22nd ACM SIGKDD*. ACM, 2016, pp. 1555–1564.

[13] C. Lee, T. Kumano, T. Matsuki, H. Endo, N. Fukumoto, and M. Sugawara, "Understanding storage traffic characteristics on enterprise virtual desktop infrastructure," in *ACM SYSTOR*, 2017, p. 13.

[15] Y. Li, H. Liu, W. Yang, D. Hu, X. Wang, and W. Xu, "Predicting inter-data-center network traffic using elephant flow and sublink information," *IEEE trans. on Network and Service Management*, vol. 13, no. 4, pp. 782–792, 2016.

[16] A. Saha, B. Samanta, N. Ganguly, and A. De, "Crpp: Competing recurrent point process for modeling visibility dynamics in information diffusion," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. ACM, 2018, pp. 537–546.

[17] F. Radlinski and T. Joachims, "Query chains: learning to rank from implicit feedback," in *SIGKDD*, 2005, pp. 239–248.

[18] C. Barakat, P. Thiran, G. Iannaconne, C. Diot, and P. Owerzazki, "Modeling internet backbone traffic at the flow level," *IEEE Trans. on Signal Processing*, vol. 51, pp. 2111–2124, 2003.

[19] F. Dong, K. Wu, and V. Srinivasan, "Copula analysis of temporal dependence structure in Markov Modulated Poisson Process and its applications," *ACM TOMPECS*, vol. 2, no. 3, p. 14, 2017.

[20] F. Ciucu, F. Poloczek, and J. Schmitt, "Sharp per-flow delay bounds for bursty arrivals: The case of FIFO, SP, and EDF scheduling," in *IEEE INFOCOM*, 2014, pp. 1896–1904.

[21] H. Z. Moayedi and M. Masnadi-Shirazi, "ARIMA model for network traffic prediction and anomaly detection," in *ITSim*, vol. 4. IEEE, 2008, pp. 1–6.

[22] M. E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: evidence and possible causes," *IEEE/ACM trans. on netw.*, vol. 5, no. 6, pp. 835–846, 1997.

[23] G. Terdik and T. Gyires, "Lévy flights and fractal modeling of internet traffic," *IEEE/ACM trans. on netw.*, vol. 17, no. 1, pp. 120–129, 2009.

[24] P. Borgnat, G. Dewaele, K. Fukuda, P. Abry, and K. Cho, "Seven years and one day: Sketching the evolution of Internet traffic," in *IEEE INFOCOM*, 2009, pp. 711–719.

[25] I. Nevat, D. M. Divakaran, S. G. Nagarajan, P. Zhang, L. Su, L. L. Ko, and V. L. Thing, "Anomaly detection and attribution in networks with temporally correlated traffic," *IEEE/ACM trans. on netw.*, vol. 26, no. 1, pp. 131–144, 2018.

[26] H. Li, K. Ota, and M. Dong, "Learning IoT in edge: deep learning for the internet of things with edge computing," *IEEE Network*, vol. 32, no. 1, pp. 96–101, 2018.

[27] B. Samanta, A. De, and N. Ganguly, "STRM: A sister tweet reinforcement process for modeling hashtag popularity," in *IEEE INFOCOM*. IEEE, 2017, pp. 1–9.

[28] B. Mao, Z. M. Fadlullah, F. Tang, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "Routing or computing? the paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE trans. on Computers*, vol. 66, no. 11, pp. 1946–1960, 2017.

[29] F. Liu, J. Guo, X. Huang, and J. C. Lui, "eBA: Efficient bandwidth guarantee under traffic variability in datacenters," *IEEE/ACM trans. on netw.*, vol. 25, no. 1, pp. 506–519, 2017.

[30] T. Joachims, "A support vector method for multivariate performance measures," in *ICML*, 2005.

[31] Netresec. (2018) Publicly available pcap files. [Online]. Available: https://www.netresec.com/index.ashx?page=PcapFiles

[32] S. Kim, X. Wang, H. Kim, T. Kwon, and Y. Choi, "CRAWDAD dataset snu/bittorrent (v. 2011-01-25)," Downloaded from https://crawdad.org/snu/bittorrent/20110125/tcpdump, Jan. 2011, traceset: tcpdump.

[33] S. Xiao, M. Farajtabar, X. Ye, J. Yan, X. Yang, L. Song, and H. Zha, "Wasserstein learning of deep generative point process models," in *NIPS*, 2017, pp. 3250–3259.

[34] P. Bao, H.-W. Shen, X. Jin, and X.-Q. Cheng, "Modeling and predicting popularity dynamics of microblogs using self-excited hawkes processes," in *WWW*, 2015.

[35] Y. Ogata, "On lewis' simulation method for point processes," *IEEE Trans on Information Theory*, 1981.

[36] S. Mishra, M.-A. Rizoiu, and L. Xie, "Modeling popularity in asynchronous social media streams with recurrent neural networks," *arXiv preprint arXiv:1804.02101*, 2018.

[37] J. H. Zar, "Spearman rank correlation," *Encyclopedia of Biostatistics*, vol. 7, 2005.