Predictive Analysis in Network Function Virtualization

Zhijing Li
zhijing@cs.ucsb.edu
UC Santa Barbara

Zihui Ge, Ajay Mahimkar, Jia Wang
{gezihui,mahimkar,jiawang}@research.att.com
AT&T Labs Research

Ben Y. Zhao, Haitao Zheng
{ravenben,htzheng}@cs.uchicago.edu
University of Chicago

Joanne Emmons, Laura Ogden
{joanne.emmons,laura.ogden}@att.com
AT&T

ABSTRACT

Recent deployments of Network Function Virtualization (NFV) architectures have gained tremendous traction. While virtualization introduces benefits such as lower costs and easier deployment of network functions, it adds additional layers that reduce transparency into faults at lower layers. To improve fault analysis and prediction for virtualized network functions (VNF), we envision a runtime predictive analysis system that runs in parallel with existing reactive monitoring systems to provide network operators timely warnings against faulty conditions. In this paper, we propose a deep learning based approach to reliably identify anomaly events from NFV system logs, and perform an empirical study using 18 consecutive months in 2016-2018 of real-world deployment data on virtualized provider edge routers. Our deep learning models, combined with customization and adaptation mechanisms, can successfully identify anomalous conditions that correlate with network trouble tickets. Analyzing these anomalies can help operators to optimize trouble ticket generation and processing rules in order to enable fast, or even proactive actions against faulty conditions.

CCS CONCEPTS

• **Networks** → **Network reliability**;

ACM Reference Format:
Zhijing Li, Zihui Ge, Ajay Mahimkar, Jia Wang, Ben Y. Zhao, Haitao Zheng, and Joanne Emmons, Laura Ogden. 2018. Predictive Analysis in Network Function Virtualization. In *2018 Internet Measurement Conference (IMC '18), October 31-November 2, 2018, Boston, MA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3278532.3278547

Keywords
Network Function Virtualization; Machine Learning

1 INTRODUCTION

Recent deployments of Network Function Virtualization (NFV) architectures [1] have gained tremendous traction. NFV allows network functions previously handled by hardware to be implemented as software running on commodity servers. Its advantages include simplifying deployment of new functionality, easier management through hosted VMs, and lower costs from using commodity hardware. The downsides are that 1) today's newly implemented virtualized network functions (VNFs) and their host commodity servers are more failure prone than dedicated hardware [11, 12, 23], and 2) virtualization introduces more layering and less visibility into lower layer events, *e.g.* faults. These downsides might negatively impact NFV deployment. For example, a critical question for NFV systems is whether they can provide availability similar to that of traditional carrier-grade systems, with up to five 9s (99.999% of uptime) [5].

In this paper, we describe our efforts to predict network failures and reduce downtime on one of the largest known NFV deployments to date, deployed on the edge of IP backbone network of a large ISP in the US. We focus on one of the important VNF types - vPE (virtualized Provider Edge router). We explore the design and performance of a system that would allow us to identify potential signatures for predicting trouble tickets in near-real-time, by applying a combination of deep learning models (LSTMs), model customization and sharing via transfer learning to syslogs.

While applying machine learning (including deep learning models) to failure prediction itself is not new [22, 28, 37], our work faces a unique combination of three challenges. First, because failures are relatively rare, our data is extremely imbalanced, making it very difficult to train a supervised learning model for fault ticket prediction. Second, since each VNF has its own specific configuration and traffic characteristics, it is likely that no single model will work well across VNFs. Third, periodic software updates constantly alter system functionality and traffic characteristics on the data plane. Thus we do not have the luxury of collecting a large training set to build a model for long term use. Instead, models must be built quickly using short windows of data, and deployed before they are made obsolete by the next software update or configuration change.

Our solution includes several techniques as follows:

- To address the data imbalance, we use an unsupervised anomaly detection approach to train a Long Short-Term Memory (LSTM) network [14] model with "normal" logs. Abnormal log patterns trigger predictions of network faulty conditions.
- To address VNF diversity, we use clustering to identify VNFs with similar configuration and log behaviors, and aggregate them (treat them as a single unit with the combined syslogs).
- To address the temporal dynamics of infrastructure changes, we use incremental training that resembles transfer learning. This helps us to quickly bootstrap a model after software updates, without incurring extended delays for collecting training data.

IMC '18, October 31-November 2, 2018, Boston, MA, USA

ACM ISBN 978-1-4503-5619-0/18/10…$15.00
https://doi.org/10.1145/3278532.3278547

We evaluate our methodology using network trouble tickets collected over a 18-month period on vPE routers deployed in production environments. Our evaluation results demonstrate that syslog anomalies often occur before network trouble tickets are generated. We can filter through these anomalies to identify any potential early warning signals or predictive signatures.

2 RELATED WORKS

Reliability and Fault Management in NFV. [9, 30] addressed the necessity and challenges of reliability, resiliency and fault management in NFV, showing that one of the key challenges is the cooperation and latency between layers. [18] studied the correlation among network resource alarms and produced rules for root cause analysis. [21, 24] leveraged Self-Organizing Map (SOM)-based clustering to identify different types of network failures based on SNMP measurements, but requires sufficient samples of each failure type in advance. [31] collected metrics from both hypervisor and VM layers, and applied Random Forest to classify VNF behaviors. All of these evaluated small-scale, self-defined network failures.

Trouble Prediction/Detection in Networking. While existing works [16, 20] achieve trouble detection based on Key Performance Indicators (KPIs), such as CPU utilization and packet loss, our work focuses on VNF syslogs. The majority of existing works apply supervised trouble prediction/detection, by building binary classifiers that are trained with both normal and abnormal events. [10, 19, 29] applied simple failure prediction methods based on characteristics of failure events, and developed Hidden Markov Model (HMM) and shallow machine learning approaches for network failure prediction. To capture sequential patterns in the monitoring data, [37] designed sequential features and applied Random Forest to learn omen and non-omen patterns for switch hardware failures in data centers. [36] applied LSTM to detect a single type of failure for server cluster down. The key challenge faced by the above supervised methods is that they require sufficient anomalous data to train the model, which takes a significant amount of time to collect, *e.g.* multiple years according to the above studies.

To reduce data collection latency, several works resorted to unsupervised approaches. [35] extracted features on state variables and identifiers, and applied PCA to perform anomaly detection. [8, 17] applied LSTM on network intrusion detection for Linux system calls and OpenStack experiments on CloudLab. While we also take an unsupervised learning approach, our work differs from existing works by focusing on predictive analysis of failures in NFV systems.

3 INITIAL ANALYSIS

Using data from a real-world NFV deployment, we study different types of network failures and their spatial and temporal patterns. We also examine patterns of the syslogs at the VNF layer, which we will use to predict network failures.

3.1 Datasets

Our dataset includes both network trouble tickets and VNF syslogs collected from 38 vPEs (virtualized Provider Edge routers), deployed by a tier-1 ISP's backbone network, over a time period of 18 months. vPE degradation can cause service impairments on customer networks. Predicting these trouble events allows operators or closed-loop automations to trigger mitigation actions prior to each event and help minimize its impact.

Network Trouble Tickets. Trouble tickets capture actionable network events. Each ticket includes the time of occurrence, the root cause, and the ticket duration. Our dataset includes the entire set of trouble tickets at these 38 vPEs, with the following six categories of root causes:

- *Maintenance*: expected or scheduled network actions or changes;
- *Circuit*: connection between two devices (on specific interfaces) is down.
- *Cable*: cable disconnection due to environmental or human artifacts.
- *Hardware*: failures of cards that constitute the chassis system and components that constitute a card.
- *Software*: failures due to software issues.
- *Duplicate*: follow-up failures when the original issue is not resolved.

For each trouble ticket, we track both the ticket report time and the repair finish time. Trouble tickets are triggered by signals from various network monitoring systems matching against known problem signatures, via a series of ticket processing logic, such as pattern matching and event correlation. Thus the ticket report time is often at or after the first occurrence of a symptom of the network fault. Since the ticket generation process is imperfect, it may miss early symptoms and introduce significant delays between the first occurrence of symptoms and the actual generation of a ticket.

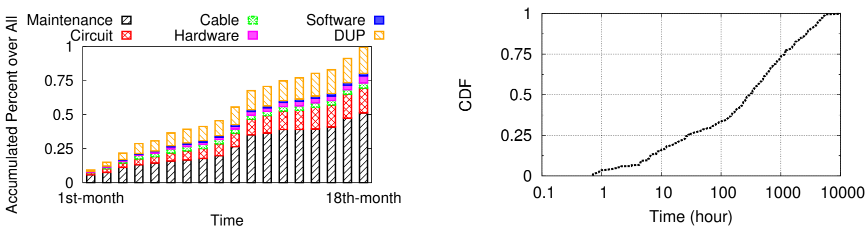
VNF Syslog. Syslogs are complex, unstructured, free-form texts generated by the systems to describe a wide range of events [26, 35]. One vPE could have millions of syslog messages per year. Both keywords and relationships among different types of log messages [8, 17, 26, 37] define the key structural patterns of syslogs. We use the well-known *Signature tree* [26] approach to transform raw syslogs into a structured representation for convenient relationship modeling.

We also compare our vPE syslogs to those of pPEs (physical Provider Edge routers) with similar number of network tickets. We observe that vPE syslogs have 77% less volume than pPE syslogs, and contain many fewer log messages on physical layer. This confirms our intuition that NFV reduces each vPE's visibility of lower layer events.

3.2 Trouble Ticket Analysis

To help understand the predictability of trouble tickets, we focus our analysis on (1) ticket temporal distribution/frequency and (2) similarity of ticket patterns between vPEs.

Temporal Distribution. Figure 1(a) shows tickets with different root causes over time. We found that maintenance is the dominant factor, but they are predictable (since they are prescheduled events). Duplicated tickets and circuit tickets are the next two major contributors. Overall, the ticket data is highly skewed. Figure 1(b) plots the distribution of inter-arrival time of non-duplicated tickets per vPE. We see that non-duplicated tickets arrive more than 40 minutes apart. 80% of time gaps between consecutive tickets are longer than 10 hours, and 25% of gaps between consecutive tickets are longer than 1000 hours (42 days). Finally, we observe that duplicated tickets often arrive in bursts.



(a) Percent of types over time (monthly).

(b) Non-duplicated ticket inter-arrival time.

Figure 1: Ticket analysis of aggregated vPEs.

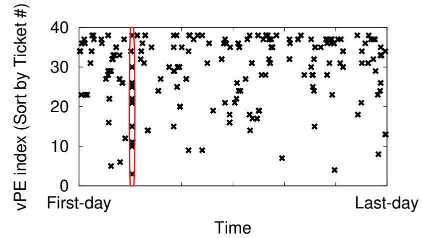


Figure 2: Tickets distributed across time (18 months) and vPEs.

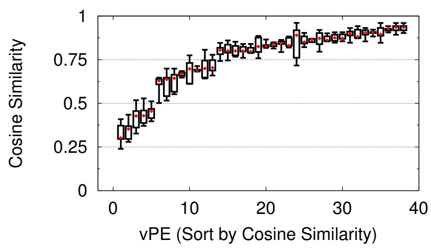


Figure 3: Cosine similarity of syslog distribution between all vPEs and individual vPE.

Per vPE Ticket Behaviors. Figure 2 shows non-maintenance trouble tickets across vPEs (sorted by their ticket volume per vPE). Each point indicates that the corresponding vPE (y) has ticket on a given time interval (x). Clearly the ticket pattern is non-periodic and vPE-dependent – a few vPEs have far more tickets than others. There is no obvious bias in time or towards any specific vPE. Another observation is that sometimes, multiple vPEs experience network faulty conditions in the same time interval (marked by the vertical bar). A deeper look at the data showed that these tickets are triggered by issues of core routers that led to disruptions at all attached vPEs. However, such cases are very rare, and only contribute to a very small percentage of trouble tickets.

3.3 VNF Syslog Analysis

We perform temporal and spatial analysis on VNF syslogs collected at vPEs. To analyze "normal" syslog entries unrelated to network failure events, we prune the log to remove any entries that are within 3 days of a ticket's active period (the period between a ticket's arrival time to when it is marked as resolved).

Correlation across vPEs. We first ask the question: do vPEs' syslogs display similar behaviors during normal operations (*i.e.*, no failures)? We compute the cosine similarity [32] of syslog distributions for each vPE v, and that of the aggregated syslog over all vPEs V, *i.e.*, $\frac{\Sigma_{i=1}^{n} s(v)_i s(V)_i}{\sqrt{\Sigma_{i=1}^{n} s(v)_i^2}\sqrt{\Sigma_{i=1}^{n} s(V)_i^2}}$, where $s(\cdot)$ denotes the syslog distribution. We use a sliding time window of one month across syslogs, and calculate the normalized frequency distribution.

Figure 3 shows the quantile values (0%, 25%, 50%, 75%, 100%) of the cosine similarity across time. Only one third of vPEs have a similar syslog distribution (cosine similarity > 0.8), and there are 5 vPEs that have < 0.5 in cosine similarity. This indicates that syslog patterns vary across vPEs, possibly due to differences in server roles, configurations and traffic. Therefore, we will need per-vPE customized models to detect anomalies on vPE syslogs.

Impact of System Updates. Another key finding is that some vPEs' syslogs had sudden changes between late 2017 and early 2018, triggered by system updates that changed the syslog distribution. We compute the cosine similarity of syslog distributions between consecutive months. We found that before the system updates, cosine similarity is consistently above 0.8, but drops below 0.4 following a system update. This means that we need to update models of vPE syslogs quickly (using short windows of data), so that they do not become obsolete.

4 PREDICTING TICKETS FROM SYSLOG ANOMALIES

In this section, we describe our effort to identify specific (or anomalous) patterns in vPE syslogs that may potentially serve as early detection or warning signatures for (trouble) ticketing conditions.

4.1 Methodology

Our empirical analysis in §3 identifies three key challenges for predicting trouble tickets via vPE syslogs. *First*, trouble tickets are relatively rare across our vPE syslogs. With such imbalanced data, it is very difficult to train a supervised learning model for fault prediction. *Second*, the volume and complexity of syslog data make it difficult to manually select the feature set necessary to train ML models on log behavior. *Third*, since syslog distributions vary across vPEs and over time, we need to customize machine learning models for each vPE, and re-train them after system updates. Both can lead to large overheads in terms of data collection delay.

To address the first two challenges, we build a Long Short-Term Memory (LSTM) network [14] that *automatically* learns syslog patterns during normal operations (§4.2). Instead of supervised training, we take an anomaly detection approach using a baseline model trained using "normal" syslog data. Thus we are unaffected by the rarity of trouble ticket events. Each detected anomaly can potentially serve as an indicator for network faulty conditions. To address the third challenge of data collection latency, we apply both clustering and online learning techniques to reduce the amount of training data required to customize models for individual vPEs (§4.3).

After detecting anomalies, we associate a mapping between them and relevant trouble tickets. We define a time window ahead of the ticket generation as the *predictive period*, and the time between ticket report and repair finish as the *affected period*. As shown in Figure 4, if an anomaly is detected during the predictive or affected

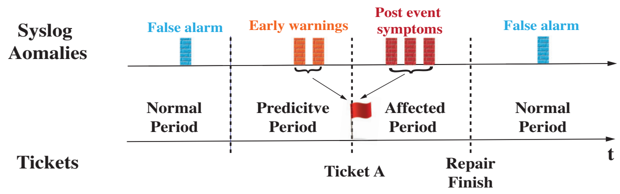


Figure 4: Mapping syslog anomalies to trouble tickets.

period of a ticket, we associate that anomaly to the ticket. Specifically, an anomaly detected during the prediction period of a ticket is treated as an "early warning signal," and those detected during a ticket's affected period are treated as "post event symptoms." Although there are many reasons why anomalies may occur before the ticket time, some of the early warning signals may be converted into alternative ticket-triggering signatures. Anomalies which are not associated with tickets will be treated as false alarms. We vary the length of the predictive period to see performance changes in Section 5.

4.2 LSTM-based Anomaly Detection

As a language for communication between users/programs and the system, vPE syslogs display sequential patterns. An accurate model of syslogs must be able to capture those sequential patterns. Thus we consider the Long Short-Term Memory (LSTM) network, which is well-known for its capability of capturing the comprehensive and intricate patterns embedded in sequential data[1]. With sufficient training data, LSTM can automatically learn normal patterns of syslogs, and flag deviations from the norm as anomalies. In fact, LSTMs have demonstrated great success in detecting a wide range of anomalies, such as server faults in distributed systems or anomalies in sentiment analysis [8, 17, 33].

Unlike traditional linear classifiers, our approach does not rely on feature engineering. For the input of LSTM, we use each individual log m_i, which captures system events for a specific interval ($[t_i, t_{i-1})$) (m_i appears at t_i). Instead of using just raw log entries, we apply the aforementioned signature tree approach [26] to extract and categorize a specific template (or signature) from the raw data, marked by a tuple of $(m_i, t_i - t_{i-1})$, $m_i \in S$, where S is the template collection. Given k syslog tuples, we train our LSTM model to predict m_{k+1}. This is a multi-class classification problem where the output is a probability distribution over the template set S.

Model Training. We train the LSTM network using syslogs produced during "ticket-free" network operations. As mentioned in Section 3.3, we prune syslog entries that occur within a 3-day buffer around the active window of actual tickets. We also experimented with larger window sizes but did not observe notable differences.

Detecting Anomalies. Using a trained LSTM model, we detect an anomaly as follows. To determine whether an incoming syslog m_{k+1} is normal or abnormal, we plugin the previously observed k syslogs into the model and derive the probability distribution of prediction of the $(k+1)$th log. If m_{k+1} is normal, then the corresponding log-likelihood value should be high (above a threshold), and abnormal if not. By changing the threshold value, we can derive a precision-recall curve (PRC), which is the most widely used measure to evaluate anomaly detection systems [6].

Learning Minority Syslog Patterns. While LSTMs are designed to automatically learn patterns of normal syslog entries, minority patterns are generally hard to learn given their rare appearances in the training data. The result is a high false alarm rate. We address this by over-sampling the minority (normal) patterns [4]. Specifically, we use month i's syslog to train a LSTM model that will be used to detect anomalies during month $(i+1)$. We apply the LSTM model training in multiple rounds, using month i's normal syslog as training data. After each round of training, we test the model using the original training data and identify normal syslog patterns that are misclassified as anomalies. We then over-sample these patterns and randomly sample all other patterns, and use the resulting data to adjust the model weights. The process exits when the false positive rate cannot be further improved.

4.3 Customization and Adaptation

Since the syslog distribution varies across vPEs, a general LSTM model will likely achieve suboptimal accuracy. The ideal solution is to build a customized model per vPE, but the resulting training overhead and data collection latency are unacceptable. We address this tradeoff between model accuracy and data collection latency using vPE grouping [16]. We apply K-means [13] to group vPEs and choose the number of groups K based on modularity. vPEs in the same cluster show similar patterns in syslog distributions, and their training data will be aggregated together to build a unified model for the group. For our dataset, we produced 4 vPE clusters, which led to 4 LSTM models.

We also reduce the latency of training data collection using online (or incremental) learning. Specifically, each month we perform a round of incremental training by updating the model weights using the newly arrived syslog entries. Since the syslog distribution is relatively stable, we do not observe significant changes in model weights.

The exception is that between late 2017 and early 2018, the vPE network had a system upgrade, and some vPEs' syslog distributions were significantly modified. As a result, the number of false alarms increased by a factor of 14, indicating that the model is out-of-date and required updating. The naive solution is to retrain the entire model, but rebuilding a reasonable training dataset takes more than 3 months. We want a solution that can retrain the models in a much shorter time window.

To address this challenge, we consider transfer learning [27], where a pretrained neural network model (*i.e.* a "teacher model" that was trained before the system update) is adapted using limited training data to a student model that can respond to new syslog behavior. Specifically, we build the student model by first copying the teacher model, then training the student model using new syslog data to fine tune the top layers of the model. For our cases, one week of new training data is sufficient to quickly update the model after a major software update.

5 EVALUATION

In this section, we evaluate our LSTM-based anomaly detection system, and the feasibility of using vPE syslog anomalies as (early) warning signatures of network trouble tickets.

[1] LSTM is a special case of Recurrent Neural Networks (RNN). It is equipped with explicit memory cells that have the ability to remember long-term dependencies over sequences. [14] provides a detailed tutorial of LSTM.

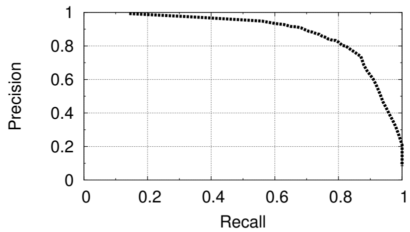


Figure 5: PRC for 1-day predictive period.

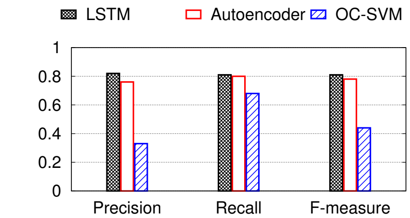


Figure 6: Anomaly detection performance of different approaches.

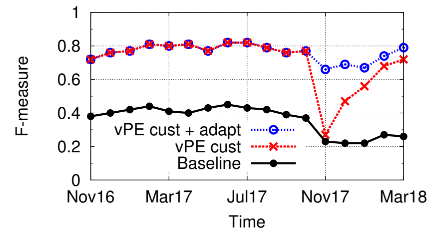


Figure 7: Effectiveness of different components.

5.1 Experimental Setup

We implemented our anomaly detection system using Keras [2] with Tensorflow [3] as the backend. For model optimization, we varied model parameters to minimize the categorical cross entropy [15], but found that model performance is generally insensitive to parameter choices. Our final LSTM model consists of 2 LSTM layers and 1 dense layer.

Estimating Ground Truth of Syslog Anomalies. Evaluation of our anomaly detection system requires ground truth of syslog anomalies, which we approximate using trouble tickets. For each trouble ticket, we define a time window before its generation time as the *predictive period*, and the time window after its generation till the reported ticket repair time (ticket duration) as the *affected period*. As shown by Figure 4, if any syslog anomaly falls into the predictive period or the affected period of a ticket, we will treat it as a true anomaly. So one ticket can possibly have multiple (early) signatures. On the other hand, any anomaly outside of these periods is treated as a false positive. We tried multiple values of predictive periods, from 1 hour to 2 days, and found that the detection performance converges at 1 day.

Another interesting observation is that after matching syslog anomalies with non-duplicated tickets, each ticket is associated with at least two anomalies (in the predictive period). These anomalies are close to each other, less than 1 minute apart on average. Thus we configure the detection system to report a warning signature for network trouble tickets upon detecting a small cluster of two or more anomalies.

Training and Testing. We use syslog data from the first month of the 18-month data for initial model training. At the end of each month, we update the LSTM model using fresh data from the previous month, and test the updated model using data from the subsequent month. Both initial model training and monthly model updates complete in less than one hour.

5.2 Accuracy of Anomaly Detection

Precision, Recall, F-Measure. We start with three standard metrics on anomaly detection [25]. Precision shows the percentage of true anomalies among all anomalies detected; Recall measures the percentage of anomalies in the data set (tickets as the ground-truth) being detected; and F-measure is the harmonic mean of the two.

Figure 5 plots the Precision-Recall Curve (PRC) produced by adjusting the aforementioned threshold in LSTM log probability (§4.2). Our final operating point is the one that maximizes the F-measure, with precision at 0.8 and recall at 0.81. In this case, our system can effectively identify anomalies while achieving low false positives at 0.6 per day for all vPEs.

Comparison to Existing Methods. We consider two existing methods on anomaly detection:

- *Autoencoder* [7] is a feed-forward multi-layer neural network in which the desired output is the input itself. After training the auto-encoder with normal data, the reconstruction error can be used as an anomaly indicator. We use the TF-IDF (term-frequency, inverse document frequency) Features [36] as the input to Autoencoder.
- *One-Class SVM* [34] uses shallow learning to build a model of the normal syslog training data, which requires feature engineering (mapping the data into a high dimensional feature space via a kernel). If a new syslog entry deviates significantly from the model, it is marked as anomaly.

For a fair comparison, we applied the same customization and adaptation mechanisms (§4.3) on all three approaches.

Figure 6 shows the performance of the three approaches. The two deep learning approaches (LSTM, Autoencoder) largely outperform the traditional classification approach (one-class SVM), because feature engineering is highly challenging given the volume and complexity of the vPE syslogs. LSTM slightly outperforms Autoencoder (a precision of 0.82 vs. 0.77), by capturing sequential patterns of the syslogs.

Gains of Customization and Adaptation. We use microbenchmarks to understand the contribution of model customization (a single model for all vPEs vs. customized models per vPE) and fast model adaptation (following a system update). Figure 7 plots the model F-measure across the 18 month period. Model customization produces significant improvement in model F-measure and precision (results not shown due to space limits). Our model adaptation component allows the system to quickly recover from disruption caused by software updates using just 1-week of training data. Using training data longer than 1 week does not produce significant improvements.

Reducing Training Overhead. Our design uses both vPE clustering and transfer learning to reduce the amount of syslog training data (for constructing and adapting the LSTM model). We evaluate their effectiveness by comparing each to their corresponding baselines. Using vPE clustering, we are able to reduce the amount of (initial) training data from 3 months to 1 month. Using transfer learning, we reduce the recover time (from software updates) from

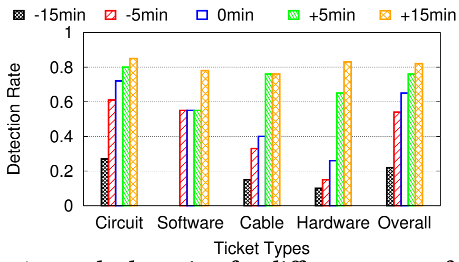


Figure 8: Anomaly detection for different types of tickets: X **time after ticket generation.**

3 months down to 1 week. This means we can build and maintain a high-quality prediction model without incurring expensive delays for collecting training data.

5.3 Trouble Ticket-based Evaluation

We use trouble tickets as approximate ground truth to evaluate how effectively our method can discover anomalous syslog conditions. Figure 8 shows the probability of detecting any anomaly related to a ticket (at least 15 minutes prior to the ticket arrival, at least 5 minutes prior, 0 minute prior, until 5 minutes after, and until 15 minutes after) for each individual (non-duplicated) ticket type, and across all the tickets.

We seek to answer the following questions:

Q1: What types of network trouble tickets show early signs in VNF syslogs?

Answer: We discover VNF syslogs appear before multiple trouble ticket types (e.g., Circuit, Software, Cable and Hardware). Syslogs related to circuit failure tickets have the highest probability of occurrence before the ticket generation (74%), followed by Software (55%), Cable (40%) and Hardware (28%). This indicates that despite reduced visibility into lower faults caused by virtualization, VNF syslogs do capture anomalous conditions related to network trouble tickets.

Q2: For failures that do not display syslog anomalies before ticket generation, will any of their anomalies show up to the syslog shortly?

Answer: Yes, for majority of tickets (80%), vPE syslogs will display anomalous patterns within 15 minutes after the ticket generation. This means that patterns of failures become visible at the NFV layer after a small delay, which can be leveraged by NFV for trouble ticket analysis, diagnosis and management.

Q3: How early do we observe syslog anomalous conditions compared to ticket generations?

Answer: The majority of detected syslog anomalies are 5 minutes ahead of the ticket generation. For Circuit, 36% of syslog anomalies are 15 minutes ahead, and the ratio is even higher for Cable (39%) and Hardware (38%) categories. Although more in-depth investigation is required, these results indicate the possibility that operators may be able to leverage these syslog anomalies to either improve their ticketing process, or identify predictive or early conditions indicative of network failures.

Q4: Can a single or group of anomalies serve as warning signatures for a group of near-term trouble tickets?

Answer: This is related to the question of whether a single syslog anomaly (or a cluster of syslog anomalies) can be associated with multiple trouble tickets. Based on our current dataset, this has never happened, mostly because the tickets are rare and well-separated. We plan to confirm this finding using larger-scale studies in the future.

Operational Findings. The anomalies identified by our model can be categorized into four scenarios. First, the detected conditions are likely true predictive signals for near-term network problems. For example, we identified a condition that involves a management daemon error message about some peer session connection failures with a particular controller (*"invalid response from peer chassis-control"*). When an anomaly with this condition was observed, it was typically followed some time later by a trouble ticket. We need to investigate this apparently predictive signature further to understand the underlying vPE behaviour. Second, the detected conditions can be analyzed and turned into early detection signatures on faulty conditions. For example, we found that a storm of protocol session flaps (*"BGP UNUSABLE ASPATH: bgp reject path"*) across multiple peers within a short time interval can be turned into a quick detection signature (with minimum false positives). This anomaly detection outperforms existing service level monitors, which normally have a longer detection delay. Third, the detected conditions could be part of the events that triggered the trouble tickets. This may be due to event response procedures in existing ticketing process flows, such as intentional delays added to suppress transient issues. Our findings may help operations to further optimize such ticketing process flows. Fourth, the detected conditions are coincidental to the ticket (*i.e.,* involving unrelated syslog anomalies). This scenario is relatively rare and should be carefully managed, *e.g.,* through adding suppression rules in ticket processing flows. In future work, we will further categorize the detected conditions into these four scenarios.

6 CONCLUSION

We use system log and network trouble tickets in a real-world deployment to study the problem of failure prediction in NFV networks. We propose a new method to detect anomalies from NFV syslogs that can potentially be used as early indicator of network issues that would typically result in trouble tickets. We validate our methodology using a sample dataset collected over 18-months on virtualized provider edge (vPE) routers in a production NFV environment. We observed that our LSTM-based anomaly detection system discovers syslog anomalous conditions that often occur before the trouble tickets. We believe our methodology can help the network operations teams to either (a) identify predictive or early warning signals, or (b) improve upon the current ticketing process that will enable timely response to NFV failures.

ACKNOWLEDGMENT

We wish to thank Jennifer Yates, Mazin Gilbert, our shepherd Dina Papagiannaki and the anonymous IMC reviewers for their useful feedback on the paper. We also wish to thank the operations teams at the ISP for helping with the data and documentation about the NFV deployment and services. This project was supported in part by NSF grants CNS-1527939 and CNS-1705042. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any funding agencies.

REFERENCES

[1] 2012. Network Functions Virtualisation. (2012). https://portal.etsi.org/NFV/NFV_White_Paper.pdf.

[2] 2018. Keras. (2018). https://keras.io/.

[3] 2018. Tensorflow. (2018). https://www.tensorflow.org/.

[4] Nitesh V Chawla. 2009. Data mining for imbalanced datasets: An overview. In *Data mining and knowledge discovery handbook*. Springer, 875–886.

[5] Daniel Collins et al. 2001. *Carrier grade voice over IP*. Vol. 2. McGraw-Hill New York.

[6] Jesse Davis and Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. In *Proc. of ICML*. ACM.

[7] Li Deng, Michael L Seltzer, Dong Yu, Alex Acero, Abdel-rahman Mohamed, and Geoff Hinton. 2010. Binary coding of speech spectrograms using a deep auto-encoder. In *Proc. of INTERSPEECH*.

[8] Min Du, Feifei Li, Guineng Zheng, and Vivek Srikumar. 2017. DeepLog: Anomaly Detection and Diagnosis from System Logs through Deep Learning. In *Proc. of CCS*.

[9] Network Function Virtualisation ETSI. [n. d.]. Resiliency Requirements, ETSI GS NFCV-REL 001, V1. 1.1. ([n. d.]).

[10] Ilenia Fronza, Alberto Sillitti, Giancarlo Succi, Mikko Terho, and Jelena Vlasenko. 2013. Failure prediction based on log files using random indexing and support vector machines. *Journal of Systems and Software* 86, 1 (2013), 2–11.

[11] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. 2015. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine* (2015).

[12] Bo Han, Vijay Gopalakrishnan, Gnanavelkandan Kathirvel, and Aman Shaikh. 2017. On the Resiliency of Virtual Network Functions. *IEEE Communications Magazine* 55, 7 (2017), 152–157.

[13] John A Hartigan and Manchek A Wong. 1979. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)* 28, 1 (1979), 100–108.

[14] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* (1997).

[15] Robert E Hoskisson, Michael A Hitt, Richard A Johnson, and Douglas D Moesel. 1993. Construct validity of an objective (entropy) categorical measure of diversification strategy. *Strategic management journal* 14, 3 (1993), 215–235.

[16] Anand Padmanabha Iyer, Li Erran Li, and Ion Stoica. 2017. Automating Diagnosis of Cellular Radio Access Network Problems. In *Proc. of MobiCom*.

[17] Gyuwan Kim, Hayoon Yi, Jangho Lee, Yunheung Paek, and Sungroh Yoon. 2016. LSTM-based system-call language modeling and robust ensemble method for designing host-based intrusion detection systems. *arXiv preprint arXiv:1611.01726* (2016).

[18] Dan Kushnir and Maayan Goldstein. 2016. Causality inference for failures in NFV. In *Proc. of INFOCOM Workshops*.

[19] Yinglung Liang, Yanyong Zhang, Anand Sivasubramaniam, Morris Jette, and Ramendra Sahoo. 2006. Bluegene/l failure analysis and prediction models. In *Proc. of DSN*.

[20] Dapeng Liu, Youjian Zhao, Haowen Xu, Yongqian Sun, Dan Pei, Jiao Luo, Xiaowei Jing, and Mei Feng. 2015. Opprentice: Towards practical and automatic anomaly detection through machine learning. In *Proc. of IMC*.

[21] Masanori Miyazawa, Michiaki Hayashi, and Rolf Stadler. 2015. vNMF: Distributed fault detection using clustering approach for network function virtualization. In *Proc. of INM*.

[22] Andrew W Moore and Denis Zuev. 2005. Internet traffic classification using bayesian analysis techniques. In *Proc. of SIGMETRICS*.

[23] Jaehyun Nam, Junsik Seo, and Seungwon Shin. 2018. Probius: Automated Approach for VNF and Service Chain Analysis in Software-Defined NFV. In *Proc. of SOSR*.

[24] Tomonobu Niwa, Masanori Miyazawa, Michiaki Hayashi, and Rolf Stadler. 2015. Universal fault detection for NFV using SOM-based clustering. In *Proc. of APNOMS*.

[25] David Martin Powers. 2011. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. (2011).

[26] Tongqing Qiu, Zihui Ge, Dan Pei, Jia Wang, and Jun Xu. 2010. What happened in my network: mining network events from router syslogs. In *Proc. of IMC*.

[27] Rajat Raina, Alexis Battle, Honglak Lee, Benjamin Packer, and Andrew Y Ng. 2007. Self-taught learning: transfer learning from unlabeled data. In *Proc. of ICML*.

[28] Maheshkumar Sabhnani and Gürsel Serpen. 2003. Application of Machine Learning Algorithms to KDD Intrusion Detection Dataset within Misuse Detection Context.. In *MLMTA*. 209–215.

[29] Felix Salfner and Miroslaw Malek. 2007. Using hidden semi-Markov models for effective online failure prediction. In *Proc. of SRDS*.

[30] Ramachandran Sathyanarayanan. 2016. Reliablity, Resiliency and Fault Management in Network Function Virtualization. *ICN* (2016), 102.

[31] Carla Sauvanaud, Kahina Lazri, Mohamed Kaâniche, and Karama Kanoun. 2016. Anomaly detection and root cause localization in virtual network functions. In *Proc. of ISSRE*.

[32] Michael Steinbach, George Karypis, Vipin Kumar, et al. 2000. A comparison of document clustering techniques. In *KDD workshop on text mining*, Vol. 400. 525–526.

[33] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*.

[34] Yanxin Wang, Johnny Wong, and Andrew Miner. 2004. Anomaly intrusion detection using one class SVM. In *Proc. of IEEE SMC Information Assurance Workshop*.

[35] Wei Xu, Ling Huang, Armando Fox, David Patterson, and Michael I Jordan. 2009. Detecting large-scale system problems by mining console logs. In *Proc. of SOSP*.

[36] Ke Zhang, Jianwu Xu, Martin Renqiang Min, Guofei Jiang, Konstantinos Pelechrinis, and Hui Zhang. 2016. Automated IT system failure prediction: A deep learning approach. In *Proc. of Big Data*.

[37] Shenglin Zhang et al. 2018. PreFix: Switch Failure Prediction in Datacenter Networks. In *Proc. of SIGMETRICS*.