CADENCE: Conditional Anomaly Detection for Events Using Noise-Contrastive Estimation

Mohammad Ruhul Amin*
moamin@cs.stonybrook.edu
Stony Brook University
Stony Brook, NY

Pranav Garg
prangarg@amazon.com
Amazon Web Services
New York, NY

Baris Coskun
barisco@amazon.com
Amazon Web Services
New York, NY

ABSTRACT

Many forms of interaction between computer systems and users are recorded in the form of event records, such as login events, API call records, bank transaction records, etc. These records are often comprised of high-dimensional categorical variables, such as user name, zip code, autonomous system number, etc. In this work, we consider anomaly detection for such data sets, where each record consists of multi-dimensional, potentially very high-cardinality, categorical variables. Our proposed technique, named Cadence, uses a combination of neural networks, low-dimensional representation learning and noise contrastive estimation. Our approach is based on estimating conditional probability density functions governing observed events, which are assumed to be mostly normal. This conditional modeling approach allows Cadence to consider each event in its own context, thereby significantly improving its accuracy. We evaluate our proposed method using both synthetic and real world data sets. Our results show that Cadence performs significantly better than existing methods at real-world anomaly detection tasks.

CCS CONCEPTS

• **Security and privacy → Intrusion/anomaly detection and malware mitigation**; • **Computing methodologies → Artificial intelligence**.

KEYWORDS

Density Estimation, Anomaly Detection, Neural Networks, Noise Contrastive Estimation

ACM Reference Format:
Mohammad Ruhul Amin, Pranav Garg, and Baris Coskun. 2019. CADENCE: Conditional Anomaly Detection for Events Using Noise-Contrastive Estimation. In *12th ACM Workshop on Artificial Intelligence and Security (AISec'19), November 15, 2019, London, United Kingdom.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3338501.3357368

*This work was done when author was with Amazon Web Services.

AISec'19, November 15, 2019, London, United Kingdom

ACM ISBN 978-1-4503-6833-9/19/11…$15.00
https://doi.org/10.1145/3338501.3357368

1 INTRODUCTION

Huge volumes of event records, such as api call records, login event records, bank transactions, function call records, etc. are being collected ubiquitously by broad range of organizations in hopes of identifying security incidents and malicious behaviors. However, recognizing malicious behavior in such data sets has proved to be a daunting task. Machine learning approaches, in particular, are yet to make consistent strides towards a solution, despite their astonishing advances in other domains, such as computer vision and natural language processing. One of the main reasons for that is lack of labelled data. That is, known malicious events are very scarce and usually not enough to train supervised methods. Even if, in a rare case, a supervised method could be trained using available labelled data, it would be unlikely to generalize to other cases, or to the same case at a different point in time, since malicious activity patterns continuously and rapidly evolve—i.e., suffer concept drift. As a result, unsupervised anomaly detection methods, where characteristics of normal events are captured from unlabelled data and outliers are marked as anomalies, remain to be a promising avenue to explore for identifying malicious events.

Motivated by this, we consider anomaly detection for data sets of event records. Our approach is based on the statistical notion that normal events are governed by an unknown probability distribution, which can be estimated using unlabeled training data [4]. Once estimated, the density function can be used at testing time to mark low-probability events as anomalies. However, low probability in a global sense does not necessarily indicate anomalies. For example, in a data set containing login event records of millions of users, a login activity in the middle of the night from an obscure browser is a very rare event—i.e., occurs with a low probability—in general, but may be quite normal for a specific user. Similarly, a globally common pattern can be rather abnormal for certain users. Therefore, it is crucial to consider each event in its own context [47]. To achieve this, instead of estimating joint probability distribution over all features, our approach focuses on estimating conditional probability distribution functions conditioned on a user-specified subset of features representing the context.

There are a few challenges in estimating probability density functions governing event records. First, most event records consist of very high-cardinality categorical features. For example, users' interactions with a web service are typically recorded by events containing features like user name, source autonomous system number, browser, operating system, user agent, etc., each taking one of hundreds of thousands, if not millions of possible values. Estimating probability density in such a high-dimensional categorical space is hard, since any realistic size of training data set is inevitably very sparse and cannot cover all possible feature combinations. In

addition, different feature values are not semantically equidistant from each other. For example, autonomous systems from the same country are semantically more similar to each other than those from different countries. Ideally, density estimation models should be able to capture such semantic relationships from training data. One way to address these challenges is to learn low-dimensional vector representations of categorical features as part of the model. Low-dimensional representation learning has shown to be very effective in modelling high-dimensional categorical data sets especially for various natural language processing tasks [36]. Recently, it has also been shown as a promising direction for detecting anomalies in high-dimensional data sets [15, 59].

Combining low-dimesional representation learning [15] with conditional anomaly detection [47], in this paper we propose a neural network model and an accompanying training procedure, called Cadence: Conditional Anomaly Detection for Events Using Noise-Contrastive Estimation. Cadence estimates conditional probability density functions for high-dimesional categorical data sets using a specific training procedure based on noise-contrastive estimation principle (NCE) [25]. In NCE, which is adopted also by [15], probability density estimation is reformulated as a binary classification task, where the goal is to discriminate observed data points from random data points sampled from a known noise distribution. But in contrast to [15], Cadence allows users to specify a set of features to represent contextual information and use an extension of NCE to estimate corresponding conditional probability density functions.

We demonstrate the efficacy of Cadence using both synthetic and real-world data sets. We use multiple synthetic data sets generated by known probability distributions with varying degrees of correlation among features to measure the accuracy of probability density estimations. Our experiments show that, Cadence estimates probability density functions more accurately than [15]. In addition, we use real-world authentication events from Los Alamos National Laboratory [28] to gauge how Cadence performs under a real anomaly detection task, where the goal is to identify a handful of known injected compromised events, which constitute approximately 0.02% of the entire test data set. Our results show that Cadence is better than several state of the art anomaly detection baseline algorithms, both in terms of the area under the precision-recall curve (PRAUC) and the precision at top K. Specifically, Cadence provides a $14-16\%$ increase in PRAUC in absolute terms and leads to $1.7-6.1$x increase in the number of positive detections, over various anomaly detection baselines.

The rest of the paper is organized as follows; In Section 2 we provide some background to noise contrastive estimation and how to extend it to estimate conditional probability density functions. We also discuss basics of low-dimensional representation learning in Section 2. In Section 3, we elaborate our method and in Section 4 we provide implemetataion details. We then present experimental results in Section 5. Following that, we discuss the limitations of our approach in Section 6. Finally, we provide related work in Section 7 and conclude in Section 8.

2 BACKGROUND

In this section we discuss the noise-contrastive estimation technique and motivate our choice of using this technique for estimating (conditional) probability density functions that form our basis for anomaly detection. We also discuss prior work on using low-dimensional distributed representations for representing discrete entities, a concept we borrow for representing categorical features that comprise events.

2.1 Probability density estimation using Noise Contrastive Estimation (NCE)

Given data $\mathcal{D}$ consisting of samples x, noise contrastive estimation (NCE) [25] is a technique to learn the underlying probability density function $p_{\mathcal{D}}(x)$. It does so by reducing the density estimation problem to a binary classification task of distinguishing the observed data points from random data points sampled from a known noise distribution.

Formally, in NCE, for each sample $x \in \mathcal{D}$, one draws K random samples x' from a known noise distribution q and then trains a binary classifier that discriminates between samples x and samples x'. The posterior probability that a sample came from the data distribution $P(D|x)$ is given by:

$$\begin{aligned} P(D=1|x) &= \frac{P(x|D=1).P(D=1)}{\sum_{D\in\{0,1\}} P(x|D).P(D)} \\ &= \frac{p_{\mathcal{D}}(x)}{p_{\mathcal{D}}(x) + K.q(x)} \end{aligned} \tag{1}$$

Assuming a family of probability distribution functions p_θ parameterized by θ, the posterior probability is:

$$\begin{aligned} P(D=1|x,\theta) &= \frac{p_\theta(x,\theta)}{p_\theta(x,\theta) + K.q(x)} \\ &= \frac{1}{1+e^{-\log(p_\theta(x,\theta))+\log(K.q(x))}} \end{aligned}$$

To fit this model to the data, similar to logistic regression [26], NCE maximizes the log-likelihood $logP(D|x,\theta)$, i.e., minimizes the cross entropy loss, over all samples. This essentially fits $\log(p_\theta(x,\theta)) - \log(K.q(x))$ to the log odds ratio and hence fits $p_\theta(x,\theta)$ to $p_{\mathcal{D}}(x)$. More specifically, Gutmann et. al [25] show that for a fixed K, under some weak assumptions on the noise distribution q, $p_\theta(x,\theta^*)$ obtained as follows:

$$\theta^* = \max_\theta \sum_{x\in\mathcal{D}} \left(logP(D=1|x,\theta) + \sum_{x'\sim q} logP(D=0|x',\theta) \right)$$

is a consistent estimate of $p_{\mathcal{D}}(x)$, and equality is achieved in the limit of large sample sizes, i.e., $p_\theta(x,\theta^*) = p_{\mathcal{D}}(x)$.

One, typically, uses standard optimization algorithms such as stochastic gradient descent [10] to learn the optimum parameters θ^* that maximize the above objective function. When a non-linear function approximator is used to model p_θ, such as in a feed forward neural network, one cannot guarantee convergence to the global optimum. However, the solution to the optimization problem converges to a local optimum that mostly works well in practice.

Noise contrastive estimation estimates the probability density function in the limit irrespective of the choice of the noise distribution q. However, in practice, for faster convergence, a good candidate for q is a noise distribution that is close to the data distribution $p_{\mathcal{D}}$. Otherwise, the discriminating task is too easy for the model to learn much structure from the data [25].

2.2 Estimating Conditional Densities Using NCE

Following [40], NCE can be used to estimate conditional probability density functions as well. Given dataset $\mathcal{D}$ consisting of samples of the form (x_1, x_2), one can estimate the conditional probability density function $p_{\mathcal{D}}(x_1|x_2)$ as follows– for each sample $x = (x_1, x_2) \in \mathcal{D}$, (1) construct a negative sample of the form $x' = (x'_1, x_2)$ by drawing x'_1 from a known noise distribution q and (2) train a binary classifier that learns to discriminate between x and x'. One can show that the posterior probability $P(D|x_1, x_2)$ is given by

$$P(D = 1|x_1, x_2) = \frac{p_{\mathcal{D}}(x_1|x_2)}{p_{\mathcal{D}}(x_1|x_2) + K.q(x_1|x_2)}$$

and a model, parameterized by θ, learns $p_{\mathcal{D}}(x_1|x_2)$ by maximizing $log\ P(D|x_1, x_2, \theta)$ on the positive and negative data. Mnih et al. [40] used this in a language modeling task to estimate $P_{\mathcal{D}}(w|c)$– the probability of the next word w in a sentence, given the context c consisting of the sequence of prior words. They used the *unigram distribution* over words, where words are sampled in proportion to their frequencies in the training dataset, as the noise distribution.

The salient aspect of NCE is that it is an unnormalized probability model, meaning that one does not need to explicitly impose the normalization condition on the learned probabilities. The optimization problem above guarantees to converge on probability density functions that are properly normalized. NCE is therefore a particularly attractive technique for modeling discrete probability density functions when the vocabulary size is large, as explicitly normalizing the probabilities for a large vocabulary is computationally very expensive [15, 16]. We choose NCE for density estimation for precisely the same reason, as opposed to adapting other techniques such as word2vec [36] for our purpose. We want to learn probability densities over a multi-dimensional, high-cardinality event space. The product space for events, or in other words the vocabulary size, can be huge, and using an unnormalized model such as NCE is a natural choice.

When NCE is used to estimate probability over a single entity such as the probability of words, using the unigram distribution as noise is extremely popular and has been shown to work well [40]. However, in a multi-dimensional setting as ours, where an event consists of multiple entities, computing the unigram distribution explodes over the exponential space and additionally, using it as the noise distribution does not work well [15]. Successfully using NCE to estimate probability densities over a multi-dimensional event space in the context of anomaly detection is one of the main technical challenges that we address in this work.

2.3 Low-dimensional Distributed Representations for Discrete Entities

Orthogonal to the choice of the technique for estimating probability densities, a question that arises when one is estimating functions over discrete entities is the question of representations. Traditional representations such as one-hot encoding, tf-idf, etc. represent discrete entities as indices in a vocabulary [23]. However, these representations have no notion of similarity between entities as each entity is treated as an atomic unit. Following Mikolov et al.'s seminal work on word embeddings [36, 37], embeddings have become very popular for representing discrete entities as vectors in a low-dimensional continuous space. Contingent on the downstream neural network models, embedding vectors learned from the data have been shown to respect relationships that exist between entities, for example, linear regularities between words in word2vec [36]. To learn relationships between different event entities that are implicit in the data, we follow the same approach. We use embedding layers in our neural network model to embed entities into a low-dimensional vector representation and use noise contrastive estimation to learn probability densities over these vector representations.

3 CADENCE: CONDITIONAL ANOMALY DETECTION FOR EVENTS USING NOISE-CONTRASTIVE ESTIMATION

In this section we describe our algorithm Cadence: Conditional Anomaly Detection for Events using Noise-Contrastive Estimation. We begin by describing the problem setting, followed by the threat model, followed by a detailed description of the algorithm.

3.1 Problem Setting

Let an event e be a collection of discrete entities that are partitioned into a set of event attributes $\mathcal{A}$ and a set of contextual attributes C. So, $e = (a, c) \in \mathcal{A} \times C$, where $\mathcal{A} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_n$ and $C = C_1 \times \ldots \times C_m$. Note that all the event attributes and the contextual attributes that describe an event are discrete and some, possibly, have a high cardinality. In this work we focus on the problem of identifying conditional anomalies for events, where the anomalous behavior is determined by the values for the event attributes $\mathcal{A}$ conditioned on the context C.

Conditional anomaly detection, introduced in [47], provides a natural definition for anomalies in many real world applications where data instances tend to be similar within a context and vary across contexts [13]. To be more specific, consider the following concrete examples:

Example 1. Consider a web service login event described as a tuple consisting of the user id, the source autonomous system number, the source IP address and the user agent. As we expect the login activity to vary from one user to another or among groups of users, anomalies in the login activity conditioned on attributes that describe the *user profile*, such as the user id, are naturally more interesting.

Example 2. As mentioned in [13], the *time of purchase* can be considered as a good choice of contextual attribute for credit card fraud detection, since purchasing patterns of users might change

throughout a year. That is, spending an exhorbitant amount during the week of Christmas might be considered normal for an individual. However, spending the same amount during a week in July will be considered anomalous, if it does not conform to the individual's normal behavior during that week.

By taking a contextual view of the data and conditioning anomalies on the user provided context, Cadence is able to detect anomalies that might not be detected if one had taken a global view. Defining a proper context for events varies for each use case and data set and requires domain knowledge. Generally speaking, the context variables shoould be chosen in such a way that event attributes follow predictable patterns given context variables.

3.2 Threat Model

Generally speaking, Cadence detects security incidents which are assocated with rare events. More specifically, Cadence can detect both external security compromises as well as insider attacks, as long as the security incident involves events that have a small likelihood of occurrence under normal operation, given the attacked context. This holds true when the attacker has limited or no knowledge about the underlying distribution that generates events for a given context or when the attacker doesn't have control over some of the event attributes.

If all the events in a security incident have a high probability of occurrence under the event distribution, it implies that the attributes we use to describe security incidents are inadequate and are unable to differentiate normal operation from attack scenarios. For example, if an attacker logs into the victim's account from a completely different autonomous system and if the source autonomous system is not one of the attributes monitored during web logins, then this compromised login event will seem completely normal as far as any anomaly detection system is concerned. Such a situation might also happen when the anomalous attack event occurs outside the scope of the events we monitor for anomaly detection. Cadence will not be able to defend against security attacks in both these scenarios.

Note, in the above discussion, we assume that the normal operation does not include abundance of security incidents that can measurably alter the event distribution. Also, we assume that the event distribution is static and does not change during testing time. In Section 6, we discuss how one can deploy Cadence in a real-world setting when these assumptions do not hold.

3.3 Cadence: Algorithmic Details

We reduce the problem of detecting conditional anomalies to the problem of learning the conditional probability distribution of events, under normal operation. So, we flag as anomalies those events that have a low probability of occurrence under the learned distribution in their corresponding contexts. This low density rejection strategy for anomaly detection has been proposed in the literature in various flavors, for detecting both point anomalies as well as conditional anomalies [11, 13, 20, 47, 48], and is a good heuristic for detecting security incidents under the above described threat model.

Formally, for an event $e = (a, c)$ where a represents the event attributes and c represents the contextual attributes, we model conditional anomalies as events with a low conditional probability $P(a|c)$. As we motivate in Section 2, estimating probability density functions such as this over a high-cardinality, sparse, discrete event space is not easy. We present Cadence, which is a novel algorithm, based on noise contrastive estimation (NCE), that learns this conditional probability distribution towards accurate conditional anomaly detection.

A naive application of NCE to learn the conditional probability distribution of events does not work in our setting, when one is estimating a probability distribution over multiple attributes. In this scenario, using the popular unigram distribution as the noise distribution in NCE does not work well and coming up with a good noise distribution warrants a separate solution [15]. In this work, however, we do not address this problem by proposing a new NCE noise distribution, as in [15]. Instead, we model the conditional density estimation problem differently that allows us to estimate the conditional probability of a single event attribute at a time using NCE, with the simple unigram noise distribution.

For an event $e = (a, c)$, where $a = (a_1, \ldots, a_n)$ and $c = (c_1 \ldots c_m)$, the conditional probability distribution $P(a|c)$ can be expressed in the following manner using Bayes' rule:

$$\begin{aligned} P(a|c) &= P(a_{1..n}|c_{1..m}) \\ &= \prod_{i=1}^{n} P(a_i|a_{i+1..n}, c_{1..m}) \end{aligned} \tag{2}$$

where, for any variable α, $\alpha_{i..j}$ denotes the set $\{\alpha_i, \alpha_{i+1}, \ldots \alpha_j\}$ when $i \leq j$ and the empty set otherwise. Note, in the right hand side, all the conditional probability distributions $P(a_i|a_{i+1..n}, c_{1..m})$ are distributions over a single event attribute a_i. Instead of learning the entire conditional probability distribution $P(a|c)$ in one shot, we learn these individual conditional probability distributions for all attributes a_i using NCE, in a joint manner, simultaneously. When we have learned all the individual conditional probabilities, their product estimates $P(a|c)$. Note that, Equation 2 does not make any assumptions and applies to all distributions. While any ordering of variables in Equation 2 is theoretically equivalent, in practice different orderings may yield different anomaly detection performances. Currently, we treat specific ordering of attributes as a hyperparamter and use basic hyper paramter optimization techniques (i.e. grid search) to figure out the best setting. We leave investigating why certain orderings work better than others in practice as future work.

Figure 1 illustrates the neural architecture used by Cadence. First, we map each input event attribute into a low-dimensional continuous representation via embeddings that are learned as part of the model training procedure. As we described in Section 2, this allows us to learn relationships between discrete event entities that are implicit in the data. Similar to field-aware factorization machines [27] and the work on word embeddings [39], we learn two different embeddings for each event attribute– the first embedding is used while estimating the conditional probability distribution of the attribute, and the second embedding is used when this attribute appears as a context while estimating the conditional probability

distribution of a different attribute. For an attribute T, let $h^T : T \rightarrow \mathbb{R}^d$ be the embedding function that embeds T when its conditional probability density function is estimated. And, let $g^T : T \rightarrow \mathbb{R}^d$ be the embedding function that embeds T when it appears as a context in the conditional probability density function for other attributes.

We learn the conditional probability distribution function over these embedded representations using NCE, as described in Section 2. To estimate $P(a_i|a_{i+1..n}, c_{1..m})$, for each positive sample $e = (a_1 \ldots a_n, c_1 \ldots c_m) \in \mathcal{D}$, we construct K negative samples of the form $e' = (a_1 \ldots a'_i \ldots a_n, c_1 \ldots c_m)$, where $a'_i \sim q_i$, where q_i is the unigram distribution over $\mathcal{A}_i$. As mentioned before, we use the embedding functions $g^{\mathcal{A}_j}$, $j \in \mathcal{A}_{i+1..n}$, and g^{C_k}, $k \in C_{1..m}$, to embed the respective attributes to create the context for attribute $\mathcal{A}_i$. We represent the combined context v_i as a weighted combination of the embedding representations of the context attributes:

$$v_i = \sum_{j=i+1}^{n} w_{ij}.g^{\mathcal{A}_j}(a_j) + \sum_{k=1}^{m} w'_{ik}.g^{C_k}(c_k)$$

And, we model the log probability $log\ P(a_i|a_{i+1..n}, c_{1..m})$ as a dot product between the context v_i and the embedding representation of attribute a_i:

$$log\ p^i_\theta(a_i|a_{i+1..n}, c_{1..m}) = v_i \odot h^{\mathcal{A}_i}(a_i)$$

Here, the parameters θ include the embedding functions g^T and h^T that are shared across all NCE models that individually estimate conditional probability densities for each event attribute, and the weights w_{ij}, w'_{ik} for each i. Our network architecture that models the conditional probability density function for each attribute a_i is based on the vector log-bilinear model in [39] that uses NCE to learn word embeddings. However, we combine these individual NCE models in a novel way that allows us to estimate conditional probability distribution functions over arbitrary discrete events consisting of multiple attributes.

Following Section 2, the posterior probability for the individual NCE discriminators is given by:

$$P_i(D = 1|a_{i..n}, c_{1..m}, \theta) = \frac{p^i_\theta(a_i|a_{i+1..n}, c_{1..m})}{p^i_\theta(a_i|a_{i+1..n}, c_{1..m}) + K.q_i(a_i)}$$

We learn the parameters θ by maximizing the posterior log-likelihood for all the discriminators in a joint manner:

$$\theta^* = \max_\theta J(\theta),\ \ where\ J(\theta) = \sum_{i=1}^{n} J_i(\theta),$$

$$J_i(\theta) = \sum_{e \in \mathcal{D}} \Big(log\ P_i(D = 1|a_{i..n}, c_{1..m}, \theta) + \sum_{e' \sim q_i} log\ P_i(D = 0|a_{i..n}, c_{1..m}, \theta) \Big)$$

Note that we learn the normalization constant for the probability distribution functions as part of the network, as in [40, 51, 62], and do not model it separately as suggested in the original work on NCE [25]. Once we obtain the optimum parameters θ^* as above, the anomaly score for an event $e = (a, c)$ is:

$$anomaly\ score(e) = -log\ P(a|c) = -\sum_i log\ p^i_{\theta^*}$$

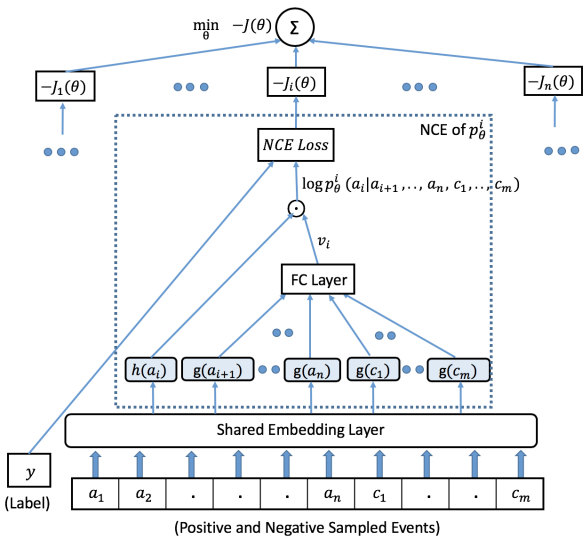


Figure 1: Neural architecture of Cadence.

4 IMPLEMENTATION DETAILS

We have implemented Cadence in Python as an MXNet module [14]. Our implementation is general to any event dataset. The user specifies the event attributes that describe an anomalous behavior and the contextual attributes that provide the context for the conditional anomalies. In addition, the user is required to provide an ordering of variables in Equation 2 that lead to equivalent, yet different, Bayes' factorizations for modeling the conditional probability distribution for events. We implemented our own logic to negatively sample attributes from their respective unigram distributions for noise-contrastive estimation. As we negatively sample, in addition to the label information, the training module receives an attribute id (*nid*) that specifies the attribute that was negatively sampled in the given sample. We equivalently express the training objective $J(\theta)$ described in Section 3.3 on a per-sample basis as follows:

$$J(\theta) = \sum_e J_e(\theta),\ \ where$$

$$J_e(\theta) = \sum_{i=1}^{n} y\ .\ log\ P_i(D = 1) + (1 - y)\ .\ \mathbb{1}[nid = i]\ .\ log\ P_i(D = 0)$$

Here, *nid* acts as a selector that selects the appropriate term $log\ P_i(D = 0)$ for each negatively labeled sample. We use the Adam optimizer [30] available in MXNet for learning the model parameters that optimize this objective function. We run all our experiments on a Tesla V100 GPU machine.

5 EXPERIMENTAL EVALUATION

In this section, we evaluate Cadence, first on a few synthetic datasets and then on real-world network authentication dataset from the Los Alamos National Laboratory (LANL) [28].

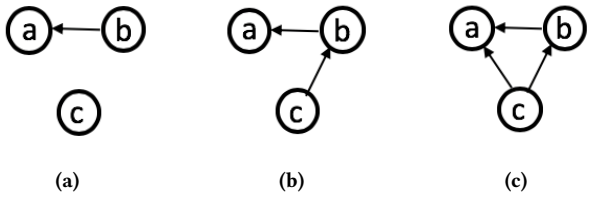


Figure 2: Bayesian network that expresses the conditional dependence structure in the (a) low dependence (b) medium dependence and (c) high dependence synthetic dataset.

5.1 Evaluation on Synthetic Datasets

First, we investigate the effects of conditional modeling and how Cadence performs under different conditions. To do that, we evaluate how well Cadence is able to identify anomalous data points in synthetic datasets generated from various probability distributions with differing complexities. To serve as a baseline, we compare Cadence with APE [15], which is an anomaly detection algorithm for events also based on noise-contrastive estimation. As opposed to Cadence, however, APE models joint probabilities over event attributies and hence is not a contextual anomaly detection method. Cadence, on the other hand, considers each event in its own context by modeling conditional probabilities given context parameters.

In the remaining of this section, we first explain how we generate synthetic datasets, then we discuss how we set hyperparamters and finally we give the details of our experiments and present results.

5.1.1 Synthetic Datasets. We synthetically create datasets by drawing samples from three different distributions that have a low, medium and high dependence between variables, as shown in Figure 2. All datasets involve three categorical variables– a, b and c. In the low dependence dataset, variable a depends only on b and b is independent of c. In the medium dependence dataset, a depends only on b and b depends only on c. Finally, the high dependence dataset has all dependencies– a depends on b and c and b depends on variable c. Both variables a and b have a cardinality 100 and their probabilities are distributed according to the power law distribution $f(x) \propto x^{-k}$, where $k = 5$. As mentioned, the distributions for a and b respect the respective dataset dependencies. We ensure this when we draw samples for a and b from the power law distribution by seeding the random seed with the values of its parent variables in the dependency graph (refer Figure 2). Variable c is uniformly distributed and has a cardinality 1000. For all the three datasets, we generate a train and test set consisting of $300K$ and $100K$ samples respectively.

Since c is uniformly distributed, the joint probability $P(a, b, c)$ is equal to the conditional probability $P(a, b|c)$ up to a constant multiplicative factor. As a result, for any sampled dataset, the rank list of events with the smallest joint probability is same as the ranklist with the smallest conditional probability. Hence, on these datasets, we can compare Cadence and APE for anomaly detection with parity, even though they estimate different probability densities to model anomalies.

5.1.2 Hyperparameter Settings. We used the following hyperparameter values for training a Cadence model on the synthetic datasets – the mini batch size is $5K$ samples, the negative sampling rate is 10 negative samples per positive sample, the learning rate is 10^{-3}, the weight decay for regularization [23] is 10^{-4}, and the embedding dimension is 128. Varying the hyperparameter values effects the performance of both Cadence and the APE algorithm in similar ways. So, in these experiments, we choose the same hyperparameter values for APE as well.

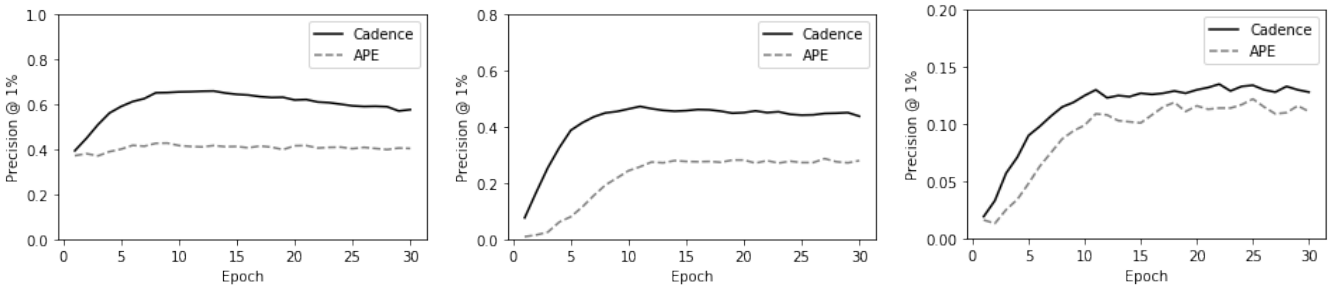
5.1.3 Detecting Anomalies. In our experiments with synthetic data, we assume that the abnormality of an event is inversely proportional to its conditional probability, $P(a, b|c)$, and the bottom 1% of events with the lowest conditional probabilities are anomalies. Based on these assumptions, we evaluate the precision in identifying these anomalies at a test bandwidth of 1%. In Figure 4, we plot Cadence's performance in terms of the *precision* @ 1% on the synthetic datasets. The precision ranges from 12% – 57% for the high to the low dependence dataset. This is significant given that anomalies consist of only 1% of all events. In comparison to APE (see Figure 3), Cadence improves the precision from 40.4% to 57.7%, 28% to 43.8%, and 11.1% to 12.8% on the low, the medium and the high dependence dataset (based on results at epoch = 30). Note, this increase in performance at anomaly detection over APE is due to our improvements in modeling the respective probability distributions. We observed similar improvements in terms of precision @ K% for other values of K.

5.2 Evaluation on the LANL Dataset

We next evaluate Cadence's effectiveness at identifying anomalies in a real-world setting using Los Alamos National Labaratories authentication dataset [28]. We also compare Cadence's performance with several state-of-the-art anomaly detection baselines. In the remaining of this section, we first give details of the dataset, then describe the experimental setup and finally present our results.

5.2.1 LANL Authentication Dataset. Los Alamos authentication dataset [28] is a labeled dataset consisting of authentication events recorded within the Los Alamos National Laboratory's internal computer network over a time period spanning around two months. In addition to the normal event activity, the dataset includes a small set of compromised authentication events that were initiated by a team of white-hat penetration testers. We compare different algorithms in terms of their effectiveness at detecting these compromised events, which form ground truth for bad behavior that is different from the normal user and computer activity in the network.

5.2.2 Experimental Setup. We describe an authentication event using the following fields present in the dataset– source user (su), source computer (sc), destination computer (dc) and authentication type (at). We split the data into a train dataset, which consists of all authentication events that occur in the network within a specific time window, and a test dataset, which consists of events (both normal as well as compromised) that occur at a time subsequent to the training window. Since the maximum number of compromised authentication events are recorded on the 9^{th} day (261/702 total compromised events), we consider a training window spanning day 1 - day 8 and test the algorithms on authentication events recorded



(a) Low dependence dataset (b) Medium dependence dataset (c) High dependence dataset

Figure 3: Performance comparison between Cadence and APE [15] at anomaly detection in the synthetic datasets.

Dataset	Training data spanning day 1 - day 8					Test data recorded on day 9		
	# records	# source users	# source comp.	# destination comp.	# auth. type	# normal events	# compromised events	% of anomalies
LANL dataset1	7.20M	1019	2707	2714	14	1.04M	261	0.025%
LANL dataset2	6.57M	1041	2635	2720	5	0.96M	261	0.027%
LANL dataset3	8.63M	1038	2630	2704	14	1.20M	261	0.021%

Table 1: Statistics about the Los Alamos authentication datasets considered in the experiments

Anomaly detection algorithm	Model at epoch = 30			Model at epoch ≤ 30 with best PRAUC		
	PRAUC	Precision @ 100 (# detections)	Precision @ 261 (# detections)	PRAUC	Precision @ 100 (# detections)	Precision @ 261 (# detections)
Los Alamos authentication Dataset 1						
Cadence	**0.300**	**0.325 (32.5)**	**0.335 (87.5)**	0.**328**	**0.350 (35.0)**	**0.376 (98.0)**
APE	0.162	0.185 (18.5)	0.210 (55.0)	0.178	0.165 (16.5)	0.164 (43.0)
DAGMM	0.013	0.000 (0.0)	0.000 (0.0)	0.035	0.000 (0.0)	0.000 (0.0)
iForest	n/a	n/a	n/a	0.070	0.030 (3.0)	0.065 (17.0)
Los Alamos authentication Dataset 2						
Cadence	**0.253**	**0.270 (27.0)**	**0.182 (47.5)**	**0.377**	**0.485 (48.5)**	**0.425 (111.0)**
APE	0.177	0.260 (26.0)	0.159 (41.5)	0.217	0.285 (28.5)	0.227 (59.0)
DAGMM	0.024	0.045 (4.5)	0.055 (14.5)	0.067	0.008 (8.0)	0.100 (26.0)
iForest	n/a	n/a	n/a	0.113	0.02 (2.0)	0.176 (46.0)
Los Alamos authentication Dataset 3						
Cadence	**0.191**	**0.055 (5.5)**	**0.185 (48.0)**	**0.244**	**0.125 (12.5)**	**0.191 (50.0)**
APE	0.101	0.040 (4.0)	0.044 (11.5)	0.104	0.040 (4.0)	0.032 (8.5)
DAGMM	0.010	0.030 (3.0)	0.011 (3.0)	0.027	0.025 (2.5)	0.013 (3.5)
iForest	n/a	n/a	n/a	0.097	0.050 (5.0)	0.080 (21.0)

Table 2: Performance at anomaly detection on the Los Alamos authentication data [28]

on day 9. We create three different smaller datasets from this data, by uniformly sampling, in each dataset, 1000 different source users and additionally include, in all the datasets, all the source users who have compromised events in the test data. This ensures that all the compromised events on day 9 are present in all the three test datasets. We perform all experiments reported in this section on these datasets. Basic statistics about these datasets is described in Table 1. Note that these training datasets consist of $6.5M - 8.6M$ events and compromised events constitute a very small fraction $0.02 - 0.025\%$ of the total events in the test datasets.

5.2.3 Anomaly Detection Baselines. For a fair evaluation of Cadence, we compare its performance with state of the art anomaly detection algorithms as baselines. The baseline methods are described below:

- *Anomaly Detection via Probabilistic Pairwise Interaction and Entity Embedding (APE) [15]*: APE uses the noise contrastive estimation technique with a "context-dependent" noise distribution to model the joint probability of events. It models this joint probability distribution by using the weighted pairwise interactions of embeddings of different entity types.

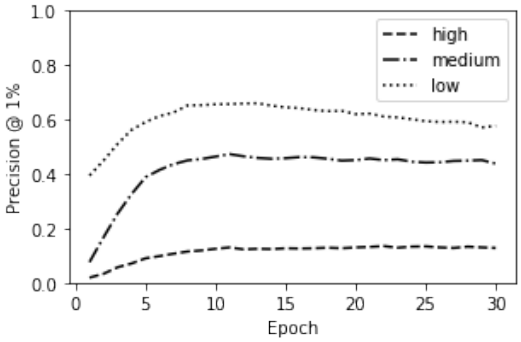


Figure 4: Cadence's performance at anomaly detection in the high, the medium and the low dependence synthetic datasets.

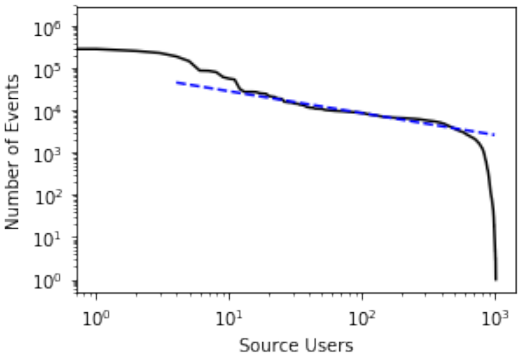


Figure 5: Number of authentication events initiated by different source users in LANL authentication dataset [28] (both axes are in log scale).

- *Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection (DAGMM) [61]*: DAGMM utilizes a deep autoencoder to obtain a low-dimensional representation and reconstruction error for each input event data that is further fed into a Gaussian mixture model. DAGMM trains by jointly optimizing the parameters of both the deep autoencoder and the mixture model, simultaneously. Events with a low probability under the mixture model are predicted as anomalies.
- *Isolation Forest (iForest) [32]*: iForest builds an ensemble of trees where each tree is fitted to a different sub-sample of the dataset and features in the dataset. For each tree, iForest isolates the anomalies by recursively partitioning the event space as it constructs the tree. Events that require fewer number of splits for isolation are predicted as anomalies.

5.2.4 Modeling Details and the Hyperparameter Settings. In the Los Alamos data, we expect the authentication activity to vary across different source users or across group of users. Figure 5 confirms this, to some extent, as the number of authentication events initiated by different source users follows an approximate

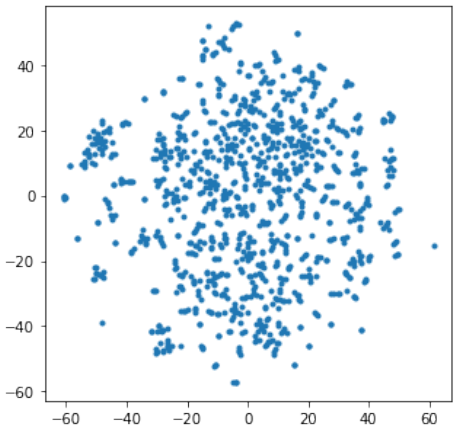


Figure 6: TSNE visualization of the embedded representation of the source users.

power law distribution (indicated by the dotted line in the figure). To factor out expected differences in behaviors of different source users, we model anomalies as $P(dc, sc, at|su)$ where su, sc, dc, and at is the source user, source computer, destination computer and the authentication type associated with an event respectively. In addition, we instantiate Cadence with the following factorization for the conditional probability density:

$$P(dc, sc, at|su) = P(dc|sc, at, su).P(sc|at, su).P(at|su)$$

The effect of varying the hyperparameter values on Cadence's performance is described in detail in Section 5.2.6. In the experiments in this section, we instantiate Cadence with a $50K$ mini batch size, negative sampling rate per attribute 10, learning rate 10^{-3}, weight decay 10^{-5} and the embedding dimension 128. We obtained these hyperparameter values by minimizing perplexity [7] on a validation set. Varying the hyperparameter values effects the performance of both Cadence and APE in similar ways. We use the same hyperparameter values for the mini batch size, the negative sampling rate and the embedding dimension for APE.

For DAGMM, we use a one-hot encoding for representing the source user, source computer, destination computer and the authentication type of an event. This results in an input vector of ~6800 dimensions for the three datasets. We instantiate DAGMM with the same network architecture that was used by the authors in [61]. We use a 1K mini batch size and 16 mixture components in DAGMM's guassian mixture model, as that gave the best results. We train all the three neural network models– Cadence, APE, DAGMM for 30 epochs.

For iForest, we use an ensemble of 100 estimators. Each tree estimator is fit on a 0.5 subsample of the training dataset, with replacement. We use a contamination rate of $1e-4$. This gave the best results for iForest and is close to the fraction of anomalies in our test dataset.

Note that training iForest is quite fast and takes less than a minute on our dataset, as opposed to Cadence , APE and DAGMM, which are essentially neural network models and their stochastic gradient descent-based training takes few hours for 30 epochs.

5.2.5 Experimental Results. We present the experimental results on the Los Alamos authentication datasets in Table 2, where we tabulate for each algorithm, the performance reported by the model obtained at epoch = 30, as well as the performance of the model obtained at an earlier epoch that has the best precision-recall characteristics. Since there is a natural class imbalance in security datasets, we measure the performance of a model in terms of the area under the precision-recall curve (PRAUC). Also, since machine learning algorithms for security are most commonly deployed with humans in the loop, we additionally report for each model its performance with respect to *precision @K* and the *number of correct detections @K*, for $K = \{100, 261\}$ (note all the three datasets have 261 known compromised events). All results in Table 2 are averages over two different runs.

We observe that Cadence detects compromised events in all three Los Alamos datasets consistently better than the baseline methods. Using Cadence to detect anomalies improves the PRAUC to 32%, 37% and 24%, in three datasets respectively, when the best performance of the models are considered. Recall that these are hard datasets with the fraction of compromised events just ranging $0.02 - 0.025\%$, hence it is very difficult to achieve high PRAUC values. Cadence improves the PRAUC over these three baselines at least by +15%, +16% and +14% on the three Los Alamos datasets.

Using Cadence to detect anomalies also improves *precision@K* values, yielding at least $1.7 - 3.1$x and $1.8 - 6.1$x more detections of compromised events at a test bandwidth $K = 100$ and $K = 261$, respectively. More specifically, in the Los Alamos datasets, of the total 1M events, top-100 most anomalous events ranked by Cadence contain $12 - 48$ compromise incidents. That is, a security operations team which investigates top 100 events ranked by Cadence would be able to confirm $12 - 48$ security incidents.

These results show that Cadence significantly outperforms the APE, DAGMM and the iForest baselines at detecting compromised events in the real-world Los Alamos dataset. Since Cadence and APE model anomalies using different probability distributions but learn these distributions using similar techniques based on NCE, our results strongly attest to the advantages of using conditional anomaly detection, for the right choice of the context.

Figure 6 shows a TSNE visualization [34] of the source users obtained from a Cadence model, trained on one of the Los Alamos datasets. The visualization confirms that our training procedure is able to learn a non-trivial distance metric between contexts. Observe that TSNE visualization reveals several small clusters consisting of a few users. As opposed to training different models for different users, which is not practically feasible, Cadence is able to take advantage of these learned distances and learn a user's probability distribution not only from its own events but also from the events of other *similar* users.

5.2.6 Effect of Hyperparameters. In Figure 7, we plot the obtained precision recall curves as we vary, one at a time, the negative sampling rate per attribute, the dimension of the vector embeddings and the mini batch size, keeping the values of the other hyperparameters fixed. Cadence reports an increase in performance as we increase the negative sampling rate and the vector dimension up to a point; the performance begins to decrease beyond negative sampling rate = 10 and vector dimension = 128. On the other hand, surprisingly, we continuously witness an improvement in its performance as we increase the mini batch size from $1K$ samples to a value as large as $100K$. Typically, in deep neural networks one prefers very small batch sizes as they lead to flat minimizers that generalize well [29]. Our results show this is not the case for Cadence. We suspect this is because our neural architecture is not very deep. In addition, even though our model has several parameters, the loss function does not exhibit lots of non-linearity. In accordance with the experimental results, we choose a fairly large value for the mini batch size– $50K$, in our experiments. A large batch size also allows us to better utilize data parallelism available on the GPU machine, effectively increasing our training throughput.

Summary: Cadence is very effective in detecting compromised events in all the three Los Alamos datasets, with a PRAUC ranging from $24\% - 37\%$, in a setting where compromised events constitute a very small fraction $0.02 - 0.025\%$ of all events. This is significantly better than the performance of other state of the art anomaly detection baselines– APE, DAGMM and the iForest algorithm. Specifically, Cadence is at least $+14 - 16\%$ better in PRAUC in absolute terms and provides $1.7 - 6.1$ times more positive detections over the APE, DAGMM or the iForest baseline.

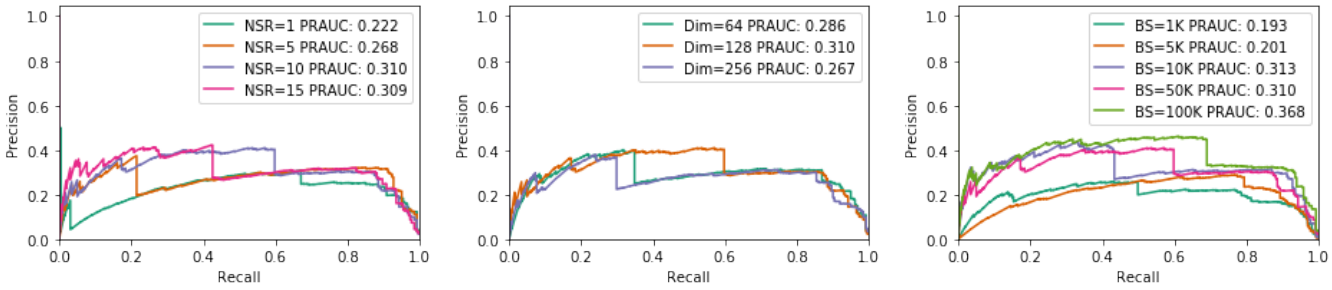
6 LIMITATIONS

Cadence is an anomaly detection method and hence suffers from the same issues that plagues other anomaly detection techniques. For example, Cadence assumes that training data comprises mostly normal events. Therefore, if the cleanliness of training data is in question, then techniques similar to those presented in [17] can be considered.

In addition, akin to any anomaly detection method in general, Cadence only detects anomalous events not malicious ones per se. While most malicious events are anomalous, the reverse is often not true and there may be numerous benign, yet rare, events in the dataset. When this happens, Cadence is likely to suffer from high false positive rates. We address this by modeling the conditional probability density function that contextualizes the events that are considered anomalies. Our results on the Los Alamos dataset show that this ameliorates this problem to some extent.

Another issue facing Cadence is that we do not model the dynamics of the event distributions. That is, we assume that the probability distribution that we learn remains static and does not evolve during testing time. Similarly, we assume that normal events do not inlcude new attribute values (out-of-vocabulary) during test time. If the event distributions are expected to change with time, Cadence needs to be retrained at frequent intervals such that the most recently trained model is in sync with the evolving distribution.

Additionally, if there are abrupt changes to the event distribution, such as those caused on a software update or on a change to the internal network topology, Cadence would label the events generated by the new distribution as false positives. However, we expect, in these situations, one can use available side information, such as the knowledge about abrupt changes to the distribution, to take preventive action, such as to retrain the Cadence model or to discard some of its anomaly predictions.



(a) (b) (c)

Figure 7: Precision-recall curves as one varies the (a) negative sampling rate per attribute (b) vector dimension (c) batch size, keeping the other hyper-parameter values fixed (the default values for the negative sampling rate = 10, vector dimension = 128 and batch size = $50K$; results obtained on the Los Alamos authentication dataset 1).

Finally, we assume there is a uniform threshold on the anomaly score irrespective of the context. However, in situations where the different contexts exhibit a huge variation in their probability distributions, one can extend CADENCE with a context-specific threshold on the anomaly scores, as well. CADENCE works under the assumption that the training data does not contain anomalies that can measurably alter the event distribution. Extending CADENCE to settings when the training data itself contains some anomalies is an interesting future direction.

7 RELATED WORK

In [13], Chandola et al. survey various anomaly detection techniques as well as their applications. Anomaly detection has been previously used in various security applications such as intrusion detection, fraud detection, fault detection etc., in a wide variety of systems such as online transactions, medical health informatics, safety critical systems, social networks, cyber security and surviellance [13, 35, 38, 44, 46].

Traditional machine learning based algorithms for anomaly detection include one-class SVM [31], nearest neighbour based anomaly detection [58], clustering based anomaly detection [19], depth-based outlier detection such as in isolation forests [32] and random cut forests [24]. Density estimation or a low density rejection strategy is another popular heuristic for anomaly detection that has been explored in various works [11, 18, 20, 47, 48]. While most algorithms mentioned above work well for continuous domains, they often break when the task consists of detecting anomalies over high-cardinality categorical features. This is because the training dataset is often very sparse. To address this issue, Aggarwal and Yu propose a subspace projection-based technique in [5], which works on high-dimensional Euclidean spaces. Additionally, in categorical spaces, the standard definitions of proximity between data points become meaningless. In the context of detecting compromised web logins, Freeman et al. [22] address this issue over the source IP by smoothing density functions over the fixed geographic hierarchy. In line with the recent success of modeling high dimensional categorical features by learning their low dimensional vector representations [36, 41], [15, 57, 61] use embeddings for dealing with sparsity while estimating densities in high-cardinality categorical datasets for anomaly detection. Embeddings also feature in other recent complementary techniques based on autoencoders [6, 59], factorization machines [60] and deep learning based algorithms that jointly optimize dimensionality reduction and clustering objectives [52, 54–56]. In contrast to the works described above, CADENCE is a conditional anomaly detection algorithm and it employs the conditional density estimation heuristic for detecting anomalies.

Conditional anomaly detection was first proposed by Song et al. [47] as a general framework for detecting those anomalies that are conditioned on a domain-specific, user-provided context. The authors argue this naturally defines anomalies in many real life applications where data instances tend to be similar within a context. Earlier works on conditional anomaly detection focused on spatial [11, 33, 45, 49] or temporal contexts [2, 3, 43], and includes regressive and auto-regressive algorithms for detecting anomalies in continuous domains [2, 3, 8, 42]. In the context of security, anomalies conditioned on user profiles were studied for security auditing, detecting credit card fraud and intrusion detection [9, 21, 50]. In these works, one essentially learns a separate anomaly detection model for each context. Song et al. [47] proposed a more refined two-step algorithm, where the first step involves clustering the contexts and the second step trains an anomaly detection model on data, conditioned on the learned clusters. However, this two-step approach is sub-optimal. In contrast, CADENCE learns a single anomaly detection model for all contexts and the training procedure, in a single step, jointly learns the contexts as well as the anomalies conditioned on the learned contexts.

For scenarios where the training data itself contains anomalies, researchers have proposed robust algorithms for clustering [1], linear regression [42], support vector machines [53], dimensionality reduction using principal component analysis (PCA) [12], and more recently deep learning based autoencoders [59]. These algorithms fit the model on the majority of the data, not all data, and in the process isolate the anomalies present in the training data. CADENCE works under the assumption that the training data does not contain anomalies that can measurably alter the event distribution. Extending CADENCE to robust settings is an interesting future direction.

The conditional probability density estimation procedure in CADENCE is based on noise contrastive estimation (NCE) [25]. NCE is

an unnormalized model to learn probability density functions. It does so by reducing the problem to binary classification between training data and samples drawn from a known noise distribution. NCE has been recently used in various NLP tasks over sentences and words such as in language modeling [40, 62] and in machine translation [51]. The anomaly detection algorithm (APE) proposed by Chen et al. [15] uses noise contrastive estimation to estimate the joint event probability and they flag an event as anomalous if it has low estimated probability. This work is very close to our algorithm. In contrast to APE, CADENCE uses NCE for detecting conditional anomalies. A comprehensive emperical comparison with APE in Section 5 shows that CADENCE outperforms APE at the task of density estimation as well as in detecting anomalies in real world settings.

8 CONCLUSION

We present CADENCE, a novel algorithm for conditional anomaly detection for events that consist of multiple high-cardinality categorical features and that abound in various security applications such as API call records, login event records, etc. CADENCE casts the conditional anomaly detection problem to one of learning the underlying conditional probability distribution that generates these events, and marks low probability events as anomalies. It achieves this by combining low-dimensional representation learning for these categorical features with a novel application of the noise contrastive estimation technique for estimating these conditional probability distributions. Our experiments on real world datasests confirm that conditioning the anomalies on appropriate contexts improves their detection. Finally, with a comprehensive experimental evaluation, we show that CADENCE is more effective in identifying compromised events than several other state of the art anomaly detection baseline algorithms.

REFERENCES

[1] 1999. ROCK: A Robust Clustering Algorithm for Categorical Attributes. In *Proceedings of the 15th International Conference on Data Engineering (ICDE '99)*. IEEE Computer Society, Washington, DC, USA, 512–. http://dl.acm.org/citation.cfm?id=846218.847264

[2] Bovas Abraham and George E. P. Box. 1979. Bayesian analysis of some outlier problems in time series. *Biometrika* 66, 2 (1979), 229–236. https://doi.org/10.1093/biomet/66.2.229

[3] B. Abraham and A. Chuang. 1989. Outlier Detection and Time Series Modeling. *Technometrics* 31, 2 (May 1989), 241–248. https://doi.org/10.2307/1268821

[4] Charu C. Aggarwal. 2013. *Outlier Analysis*. Springer. https://doi.org/10.1007/978-1-4614-6396-2

[5] Charu C. Aggarwal and Philip S. Yu. 2001. Outlier Detection for High Dimensional Data. In *Proceedings of the 2001 ACM SIGMOD International Conference on Management of Data (SIGMOD '01)*. ACM, New York, NY, USA, 37–46. https://doi.org/10.1145/375663.375668

[6] Jinwon An and Sungzoon Cho. 2015. Variational Autoencoder based Anomaly Detection using Reconstruction Probability.

[7] Leif Azzopardi, Mark Girolami, and Keith van Risjbergen. 2003. Investigating the relationship between language model perplexity and IR precision-recall measures. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*. ACM, 369–370.

[8] Ana Maria Bianco. 2001. Outlier Detection in Regression Models with ARIMA Errors Using Robust Estimates. *Journal of Forecasting* 20, 8 (2001), 565–79.

[9] Richard J. Bolton, David J. Hand, and David J. H. 2001. Unsupervised Profiling Methods for Fraud Detection. In *Proc. Credit Scoring and Credit Control VII*. 5–7.

[10] Léon Bottou. 2010. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*. Springer, 177–186.

[11] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. 2000. LOF: Identifying Density-based Local Outliers. *SIGMOD Rec.* 29, 2 (May 2000), 93–104. https://doi.org/10.1145/335191.335388

[12] Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust Principal Component Analysis? *J. ACM* 58, 3, Article 11 (June 2011), 37 pages. https://doi.org/10.1145/1970392.1970395

[13] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly Detection: A Survey. *ACM Comput. Surv.* 41, 3, Article 15 (July 2009), 58 pages.

[14] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274* (2015).

[15] Ting Chen, Lu An Tang, Yizhou Sun, Zhengzhang Chen, and Kai Zhang. 2016. Entity Embedding-Based Anomaly Detection for Heterogeneous Categorical Events. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*. 1396–1403. http://www.ijcai.org/Abstract/16/201

[16] Wenlin Chen, David Grangier, and Michael Auli. 2016. Strategies for Training Large Vocabulary Neural Language Models. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, 1975–1985. https://doi.org/10.18653/v1/P16-1186

[17] Gabriela F. Cretu, Angelos Stavrou, Michael E. Locasto, Salvatore J. Stolfo, and Angelos D. Keromytis. 2008. Casting out Demons: Sanitizing Training Data for Anomaly Sensors. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy (SP '08)*. IEEE Computer Society, Washington, DC, USA, 81–95. https://doi.org/10.1109/SP.2008.11

[18] Kaustav Das and Jeff Schneider. 2007. Detecting Anomalous Records in Categorical Datasets. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '07)*. ACM, New York, NY, USA, 220–229. https://doi.org/10.1145/1281192.1281219

[19] Lian Duan, Lida Xu, Ying Liu, and Jun Lee. 2009. Cluster-based outlier detection. *Annals of Operations Research* 168, 1 (2009), 151–168.

[20] Ran El-Yaniv and Mordechai Nisenson. 2006. Optimal Single-Class Classification Strategies. In *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 4-7, 2006*. 377–384. http://papers.nips.cc/paper/2987-optimal-single-class-classification-strategies

[21] Tom Fawcett and Foster Provost. 1999. Activity Monitoring: Noticing Interesting Changes in Behavior. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '99)*. ACM, New York, NY, USA, 53–62. https://doi.org/10.1145/312129.312195

[22] David Freeman, Sakshi Jain, Markus Dürmuth, Battista Biggio, and Giorgio Giacinto. 2016. Who Are You? A Statistical Approach to Measuring User Authenticity. In *NDSS*. The Internet Society.

[23] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.

[24] Sudipto Guha, Nina Mishra, Gourav Roy, and Okke Schrijvers. 2016. Robust Random Cut Forest Based Anomaly Detection on Streams. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 2712–2721. http://dl.acm.org/citation.cfm?id=3045390.3045676

[25] Michael Gutmann and Aapo Hyvärinen. 2010. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2010, Chia Laguna Resort, Sardinia, Italy, May 13-15, 2010*. 297–304. http://www.jmlr.org/proceedings/papers/v9/gutmann10a.html

[26] David W Hosmer Jr, Stanley Lemeshow, and Rodney X Sturdivant. 2013. *Applied logistic regression*. Vol. 398. John Wiley & Sons.

[27] Yuchin Juan, Yong Zhuang, Wei-Sheng Chin, and Chih-Jen Lin. 2016. Field-aware Factorization Machines for CTR Prediction. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16)*. ACM, New York, NY, USA, 43–50. https://doi.org/10.1145/2959100.2959134

[28] Alexander D. Kent. 2015. Comprehensive, Multi-Source Cyber-Security Events. Los Alamos National Laboratory. https://doi.org/10.17021/1179829

[29] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. 2016. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. *CoRR* abs/1609.04836 (2016). arXiv:1609.04836 http://arxiv.org/abs/1609.04836

[30] Diederik P Kingma and Jimmy Lei Ba. 2014. Adam: Amethod for stochastic optimization. In *Proc. 3rd Int. Conf. Learn. Representations*.

[31] Kun-Lun Li, Hou-Kuan Huang, Sheng-Feng Tian, and Wei Xu. 2003. Improving one-class SVM for anomaly detection. In *Machine Learning and Cybernetics, 2003 International Conference on*, Vol. 5. IEEE, 3077–3081.

[32] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*. IEEE, 413–422.

[33] Chang-Tien Lu, Dechang Chen, and Yufeng Kou. 2003. Algorithms for Spatial Outlier Detection. In *Proceedings of the Third IEEE International Conference on Data Mining (ICDM '03)*. IEEE Computer Society, Washington, DC, USA, 597–. http://dl.acm.org/citation.cfm?id=951949.952103

[34] Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.

[35] Xi Meng, Haowen Mo, Shenhe Zhao, and Jianqiang Li. 2017. Application of anomaly detection for detecting anomalous records of terroris attacks. In *Cloud Computing and Big Data Analysis (ICCCBDA), 2017 IEEE 2nd International Conference on*. IEEE, 70–75.

[36] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient Estimation of Word Representations in Vector Space. *CoRR* abs/1301.3781 (2013). arXiv:1301.3781 http://arxiv.org/abs/1301.3781

[37] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Distributed Representations of Words and Phrases and Their Compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 3111–3119. http://dl.acm.org/citation.cfm?id=2999792.2999959

[38] Robert Mitchell and Ray Chen. 2013. Behavior-rule based intrusion detection systems for safety critical smart grid applications. *IEEE Transactions on Smart Grid* 4, 3 (2013), 1254–1263.

[39] Andriy Mnih and Koray Kavukcuoglu. 2013. Learning Word Embeddings Efficiently with Noise-contrastive Estimation. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13)*. Curran Associates Inc., USA, 2265–2273. http://dl.acm.org/citation.cfm?id=2999792.2999865

[40] Andriy Mnih and Yee Whye Teh. 2012. A fast and simple algorithm for training neural probabilistic language models. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. http://icml.cc/2012/papers/855.pdf

[41] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.

[42] Peter J. Rousseeuw and Mia Hubert. 2018. Anomaly detection by robust statistics. *Wiley Interdisc. Rew.: Data Mining and Knowledge Discovery* 8, 2 (2018). https://doi.org/10.1002/widm.1236

[43] Stan Salvador and Philip Chan. 2005. Learning States and Rules for Detecting Anomalies in Time Series. *Applied Intelligence* 23, 3 (Dec. 2005), 241–255. https://doi.org/10.1007/s10489-005-4610-3

[44] David Savage, Xiuzhen Zhang, Xinghuo Yu, Pauline Chou, and Qingmai Wang. 2014. Anomaly detection in online social networks. *Social Networks* 39 (2014), 62–70.

[45] Shashi Shekhar, Chang-Tien Lu, and Pusheng Zhang. 2001. Detecting Graph-based Spatial Outliers: Algorithms and Applications (a Summary of Results). In *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '01)*. ACM, New York, NY, USA, 371–376. https://doi.org/10.1145/502512.502567

[46] Vasilios A Siris and Fotini Papagalou. 2004. Application of anomaly detection algorithms for detecting SYN flooding attacks. In *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, Vol. 4. IEEE, 2050–2054.

[47] Xiuyao Song, Mingxi Wu, Christopher M. Jermaine, and Sanjay Ranka. 2007. Conditional Anomaly Detection. *IEEE Trans. Knowl. Data Eng.* 19, 5 (2007), 631–645.

[48] Ingo Steinwart, Don Hush, and Clint Scovel. 2005. A Classification Framework for Anomaly Detection. *J. Mach. Learn. Res.* 6 (Dec. 2005), 211–232. http://dl.acm.org/citation.cfm?id=1046920.1058109

[49] Pei Sun and Sanjay Chawla. 2004. On Local Spatial Outliers. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM '04)*. IEEE Computer Society, Washington, DC, USA, 209–216. http://dl.acm.org/citation.cfm?id=1032649.1033456

[50] H. S. Teng, K. Chen, and S. C. Lu. 1990. Adaptive Real-Time Anomaly Detection Using Inductively Generated Sequential Patterns. In *Proceedings of the 1990 IEEE Symposium on Research in Computer Security and Privacy*. 278–284.

[51] Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. 2013. Decoding with Large-Scale Neural Language Models Improves Translation. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. 1387–1392. http://aclweb.org/anthology/D/D13/D13-1140.pdf

[52] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Unsupervised Deep Embedding for Clustering Analysis. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 478–487. http://dl.acm.org/citation.cfm?id=3045390.3045442

[53] Huan Xu, Constantine Caramanis, and Shie Mannor. 2009. Robustness and Regularization of Support Vector Machines. *J. Mach. Learn. Res.* 10 (Dec. 2009), 1485–1510. http://dl.acm.org/citation.cfm?id=1577069.1755834

[54] Bo Yang, Xiao Fu, and Nicholas D. Sidiropoulos. 2017. Learning From Hidden Traits: Joint Factor Analysis and Latent Clustering. *IEEE Trans. Signal Processing* 65, 1 (2017), 256–269. https://doi.org/10.1109/TSP.2016.2614491

[55] Bo Yang, Xiao Fu, Nicholas D. Sidiropoulos, and Mingyi Hong. 2017. Towards K-means-friendly Spaces: Simultaneous Deep Learning and Clustering. In *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017*. 3861–3870. http://proceedings.mlr.press/v70/yang17b.html

[56] Jianwei Yang, Devi Parikh, and Dhruv Batra. 2016. Joint Unsupervised Learning of Deep Representations and Image Clusters. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*. 5147–5156. https://doi.org/10.1109/CVPR.2016.556

[57] Shuangfei Zhai, Yu Cheng, Weining Lu, and Zhongfei Zhang. 2016. Deep Structured Energy Based Models for Anomaly Detection. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48 (ICML'16)*. JMLR.org, 1100–1109. http://dl.acm.org/citation.cfm?id=3045390.3045507

[58] Manqi Zhao and Venkatesh Saligrama. 2009. Anomaly detection with score functions based on nearest neighbor graphs. In *Advances in neural information processing systems*. 2250–2258.

[59] Chong Zhou and Randy C Paffenroth. 2017. Anomaly detection with robust deep autoencoders. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 665–674.

[60] Mengxiao Zhu, Charu C. Aggarwal, Shuai Ma, Hui Zhang, and Jinpeng Huai. 2017. Outlier Detection in Sparse Data with Factorization Machines. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, CIKM 2017, Singapore, November 06 - 10, 2017*. 817–826. https://doi.org/10.1145/3132847.3132987

[61] Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. (2018).

[62] Barret Zoph, Ashish Vaswani, Jonathan May, and Kevin Knight. 2016. Simple, Fast Noise-Contrastive Estimation for Large RNN Vocabularies. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. 1217–1222. http://aclweb.org/anthology/N/N16/N16-1145.pdf