Action Sequence Augmentation for Early Graph-based Anomaly Detection

Tong Zhao*†, Bo Ni*†, Wenhao Yu†, Zhichun Guo†, Neil Shah‡, Meng Jiang†
† University of Notre Dame, IN, USA
‡ Snap Inc., CA, USA
{tzhao2,bni,wyu1,zguo5,mjiang2}@nd.edu,nshah@snap.com

ABSTRACT

The proliferation of web platforms has created incentives for online abuse. Many graph-based anomaly detection techniques are proposed to identify the suspicious accounts and behaviors. However, most of them detect the anomalies once the users have performed many such behaviors. Their performance is substantially hindered when the users' observed data is limited at an early stage, which needs to be improved to minimize financial loss. In this work, we propose Eland, a novel framework that uses action sequence augmentation for early anomaly detection. Eland utilizes a sequence predictor to predict next actions of every user and exploits the mutual enhancement between action sequence augmentation and user-action graph anomaly detection. Experiments on three real-world datasets show that Eland improves the performance of a variety of graph-based anomaly detection methods. With Eland, anomaly detection performance at an earlier stage is better than non-augmented methods that need significantly more observed data by up to 15% on the Area under the ROC curve.

CCS CONCEPTS

• **Mathematics of computing** → *Graph algorithms*; • **Information systems** → *Spam detection*; *Social networks*; • **Human-centered computing** → *Social network analysis.*

KEYWORDS

Graph machine learning, Anomaly detection, Graph data augmentation

ACM Reference Format:
Tong Zhao*†, Bo Ni*†, Wenhao Yu†, Zhichun Guo†, Neil Shah‡, Meng Jiang†. 2021. Action Sequence Augmentation for Early Graph-based Anomaly Detection. In *Proceedings of the 30th ACM Int'l Conf. on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, Australia.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3459637.3482313

* Equal contribution.

CIKM '21, November 1–5, 2021, Virtual Event, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3482313

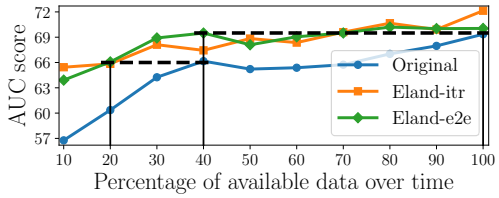


Figure 1: Performance of Dominant [7] and Eland on a social media dataset considering the *earliest* 10%-100% data from each user's action sequence. Both our Eland-itr and Eland-e2e with only 20% (40%) of available data can achieve the same performance as Dominant with 40% (100%) of data.

1 INTRODUCTION

Social networks and review platforms indirectly create a market for malicious incentives, enabling malicious users to make huge profits via suspicious behaviors, e.g., fake reviews, hijacking trending topics. Such behaviors have severe negative impact on our society. User behavior data plays an essential role in the detection of malicious users. In the databases, each user creates a sequence of actions like giving a review to a particular item. In order to leverage the homophily of users, a great line of research work has been done to construct a user-item bipartite weighted graph and develop graph-based anomaly detection algorithms such as graph embeddings and graph neural networks. The weight is for the frequency of behaviors that associate the user and item in his/her action sequence. For example, Kumar et al. [23] created a "user-reviewed-product" graph from each Amazon user's sequence of reviews; Rayana and Akoglu [36] built "user-reviewed-restaurants/hotels" graphs from Yelp users' reviewing behaviors; Zhao et al. [50] studied a "user-posted-message" graph from posting behaviors on social media.

Most of the existing work focuses on detecting anomalous users *reactively*, i.e., when their malicious behaviors have already affected many people [39]. For example, a hijacked topic might have been on the trending list on Twitter for hours, and millions of users already saw and believed it; fake reviews on Yelp could have already damaged a restaurant's reputation. Therefore, we argue that anomaly detection would be much more useful when it could be done *early*, or *proactively* to stop the malicious users before they achieve their targets. In this work, we study the problem of *early graph-based anomaly detection* when the observed behavior data is limited.

Performing anomaly detection at an early stage is challenging due to the scarcity of available observations. Despite the technical advances of existing graph anomaly detection methods, we still witness substantially reduced performance in such settings where

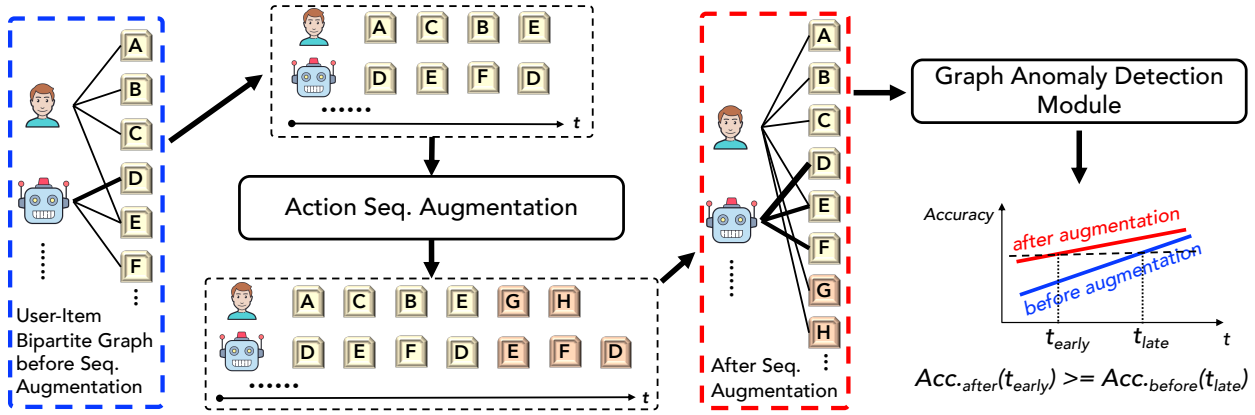


Figure 2: ELAND framework: First, the dynamic original graph is converted to action sequences, and then a sequence-based action predictor is trained to predict items that a user will adopt in the future based on his/her action history. The predicted behaviors together with the original graph form a richer, more informative augmented graph, which enables improvements upon a variety of graph anomaly detection methods. The significantly augmented data supports accurate early anomaly detection. ELAND improves the performance given equal observed data or comparable performance with less observed data.

data is insufficient or incomplete. Empirically, we observe that state-of-the-art anomaly detection methods such as DOMINANT [7] would have a relative decrease of 15% on Area under the ROC curve (AUC) when only the earliest 20% of the data (for each user) is available (see the blue curve in Figure 1). Similar degradations occur on a few other types of graph learning methods such as HETGNN [48] and DEEPAE [55] (Figure 3), leading us to ask: can we improve the performance at an early stage, when data is scarce?

Our idea is to learn and predict actions to augment the data at the early stage, and hence boost the performance of anomaly detection by "forecasting the future." Although one anomalous user might not have sufficient behaviors to be detected, detection methods could still identify him with high confidence if his likely future behaviors are provided. That is, we predict the users' future behaviors by finding patterns from the entire data and prolonging their action sequence with the items that they may adopt in the future. With the predicted actions of a large number of users, the user-item (bipartite weighted) graph can be augmented to contain much richer information than before, thus enabling the detection methods to more accurately detect anomalies from the graph data.

Present work. We propose ELAND (Early Anomaly Detection), a novel framework that achieves effective early graph anomaly detection via action sequence augmentation. ELAND has two components: (1) a sequence augmentation module that predicts the user actions and augments the graph data and (2) an anomaly detection module that detects anomalous users from graph data. We present two methods to train the proposed framework: (1) ELAND-ITR where the two components are inter-dependent on each other and hence trained iteratively in a bootstrapping style, and (2) ELAND-E2E where we jointly train the two modules as an end-to-end model. Shown in Figure 2 is the illustration of the framework. The framework enables us to take advantage of the patterns of benign and malicious users discovered by the detection module to enhance the augmentation module, and vice versa.

Figure 1 shows that (1) our ELAND (green and orange curves) that uses the users' earliest 20% data can achieve the same performance as the original method (blue curve) that uses the earliest 40%, and (2) ELAND that uses 40% data can achieve the performance of the original method with full data (100%). Such observations indicate that ELAND could save **half the time** to collect users' data for accurate anomaly detection.

The contributions of this work are summarized as follows:

- We propose a novel idea that is to achieve early-stage graph anomaly detection by action sequence augmentation. Considering behavior data as sequences, we employ sequence prediction models (e.g., Seq2Seq) to forecast the behaviors.
- We design a novel framework, ELAND, consisting of two components, action sequence augmentation and augmented graph anomaly detection to achieve early anomaly detection.
- We conduct extensive experiments on three real-world datasets. ELAND achieves better performance on both unsupervised and supervised anomaly detection methods given less (earlier) data, with up to 15% improvement on AUC score.

2 RELATED WORK

In this section we discuss three topics related to our work.

Graph Anomaly Detection has received a great amount of academic interest in the past decade [1, 8, 50, 51, 54]. Several methods [14, 23, 36, 42] were proposed following the graph outlier detection strategy. Similar approaches have been developed for bipartite graphs. With the recent advances of graph neural networks (GNN), several GNN-based anomaly detection methods [9, 12, 25, 33, 45] were proposed. DOMINANT [7] was an unsupervised attributed graph auto-encoder that detects anomalous nodes. Zhang et al. [49] proposed multi-task GCN-based model that unified recommendation and anomaly detection. DEEPAE [55] was an unsupervised graph auto-encoder-based method that detects anomalies by preserving multi-order proximity. Zhao et al. [50, 51] proposed an unsupervised loss function that trains GNN to learn user representations tailored for anomaly detection.

Sequence Prediction. Sequential data prediction is a key problem in machine learning. For example, text generation aims to predict the next word on the given context [4]; action prediction in computer vision aims to infer next action in video data [22, 26]. Recurrent neural networks (RNN) have enjoyed considerable success in various sequence prediction tasks for understanding dynamical structure of data and producing accurate prediction. Transformer models eschew recurrence and instead rely on the stacked multi-head self-attention to draw global dependencies between input tokens [43]. Recent work in recommender systems used sequence prediction to address the *cold-start problem* [35] besides many that created new graph neural models [11, 27] which ignore the sequential patterns in users' actions. In our work, we focus on graph-based anomaly detection – we predict the future actions to augment the user's action sequence, and exploit the mutual enhancement of sequence augmentation and graph anomaly detection.

Graph Neural Networks. As spectral GNNs generally operate on the full adjacency [6, 21, 29], spatial-based methods which perform graph convolution with neighborhood aggregation became prominent [16–18, 44, 52], owing to their scalability and flexibility [46]. More recently, dynamic graph learning methods [28, 31] have proposed combining GNNs with RNNs to learn on dynamic graphs. GCRN [38] proposed a modified RNN by replacing fully connected layers with GCN layers [21]. EvolveGCN [34] proposed to use GRU to learn the parameter changes in GCN instead of node representation changes. JODIE [24] proposed a user-item interaction prediction method based on historical interactions. Such works have also been applied to large-scale industrial problems [37, 41].

3 PROBLEM DEFINITION

Consider a bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E})$ at timestamp t, where $\mathcal{U}$ is the set of m users, $\mathcal{V}$ is the set of n items and $\mathcal{E}$ is the set of edges. Let $\mathbf{A} \in \mathbb{N}^{m\times n}$ be the adjacency matrix. Let $\mathbf{X}^u \in \mathbb{R}^{m\times k_u}$ and $\mathbf{X}^v \in \mathbb{R}^{n\times k_v}$ be the user feature matrix and item feature matrix, where k_u and k_v are the dimensions of raw features. As the feature dimensions of users and items are usually the same or can be projected to the same space via linear transformation, we use $\mathbf{X} \in \mathbb{R}^{(m+n)\times k}$ to denote the vertically concatenated feature matrix of $\mathbf{X}^u$ and $\mathbf{X}^v$. Let the action sequence for each user u be $x^{(u)} = \{\mathbf{x}_1^{(u)}, \ldots, \mathbf{x}_{l_u}^{(u)}\}$, in which l_u is the length of u's action sequence and each item $\mathbf{x}_i^{(u)} \in \mathbb{R}^k$ stands for the feature vector of the corresponding item node. More actions between the same user-item pair result in with edges with higher weight. We also denote $\mathbf{y} \in \{0, 1\}^m$ as labels for users where anomalies get 1 and others get 0. We follow the widely accepted definition of anomalous users in web graphs by previous works [20, 23, 50, 51]. For example, the anomalous user accounts in social networks are the botnets or the ones that frequently post advertisements or malicious links.

Following the above notations and definitions, we define the task of early-stage graph-based anomaly detection. Let $g : (\mathbf{A}, \mathbf{X}) \to \hat{\mathbf{y}}$ be a (supervised or unsupervised) graph anomaly detection method that returns a vector of prediction logits $\hat{\mathbf{y}} \in [0, 1]^m$. Let T be a later time when the observed data is "sufficient" for existing graph anomaly detection methods to perform well. We aim to design a framework that can achieve comparable or better performance at an earlier time $t < T$ when observations were "incomplete."[1] Formally, our goal is to find a data augmentation framework satisfying the following criteria:

Definition 1. *(Early-stage Graph Anomaly Detection) Let $h : (\hat{\mathbf{y}} \in \mathbb{R}^m, \mathbf{y} \in \mathbb{R}^m) \to \mathbb{R}$ be an evaluation metric (e.g., f-measure) such that a larger value is more desirable holding other conditions the same. Let g be an anomaly detection method. Design an augmenter function $f : (\mathbf{A}, \mathbf{X}) \to \mathbf{A}'$ that satisfies $h(g(f(\mathbf{A}, \mathbf{X}), \mathbf{X}), \mathbf{y}) \geq h(g(\mathbf{A}, \mathbf{X}), \mathbf{y})$. When assuming performance to increase monotonically with data, the above criteria is also equivalent to $\exists T > t$ s.t.*

$$h(g(f(\mathbf{A}, \mathbf{X}), \mathbf{X}), \mathbf{y}) \geq h(g(\mathbf{A}^*, \mathbf{X}), \mathbf{y}) \tag{1}$$

where $\mathbf{A}$ and $\mathbf{A}^*$ are the adjacency matrices at earlier time t and later time T, respectively.

When applying graph anomaly detection methods on real-world data, which usually has high complexity and uncertainty (and adversariality), it is difficult to theoretically guarantee that performance would monotonically increase with data. In the following two sections, we first introduce our proposed framework Eland which approximates $g(f(\mathbf{A}, \mathbf{X}), \mathbf{X})$, and show that Eland *empirically* satisfies the above desired property across choices of g, $\mathbf{A}$, and $\mathbf{X}$.

4 THE ELAND FRAMEWORKS

In this section, we first present the two major components of our proposed framework, Eland. Then we introduce two ways of training Eland: a bootstrapping-style iterative training Eland-itr that can be used on any existing graph anomaly detection methods; and Eland-e2e that trains both modules together in an end-to-end fashion.

4.1 Graph Anomaly Detection Module

The first component of our proposed Eland framework is a graph anomaly detection module (g). Notably, this part of Eland is general and model-agnostic, in the sense that any anomaly detection or node classification model (e.g., Dominant [7], DeepAE [55], GCN [21]) suffices and can be used. Without loss of generality, let the anomaly detection model g_{ad} be defined as:

$$\hat{\mathbf{y}} = g_{ad}(\mathbf{A}, \mathbf{X}; \Theta), \tag{2}$$

where $\hat{\mathbf{y}} \in [0, 1]^m$ is the predicted suspiciousness of the user nodes of being anomalies, and Θ denotes trainable parameters. During training, if g_{ad} is unsupervised method (e.g., Dominant [7]), its own training objective is used; if g_{ad} is supervised method (e.g., GCN [21]), we use a standard binary cross-entropy loss.

Neural-based graph representation learning methods (e.g., graph neural networks) are capable of learning low-dimensional node representations as well as making predictions. We take advantage of the learned representations of user nodes. Without loss of generality, here we take the widely used Graph Convolutional Network (GCN) [21] as an example of the anomaly detection method. The graph convolution operation of each GCN layer is defined as:

$$\mathbf{H}^{(j)} = \sigma(\tilde{\mathbf{D}}^{-\frac{1}{2}}\tilde{\mathbf{A}}\tilde{\mathbf{D}}^{-\frac{1}{2}}\mathbf{H}^{(j-1)}\mathbf{W}^{(j)}), \tag{3}$$

[1] Although we consider the terms "sufficient" and "incomplete" relatively for discussion, they can easily be made rigorous via practical constraints.

where j indicates the layer, $\mathbf{H}^j$ is the node embedding matrix generated by j-th layer, $\mathbf{W}^{(j)}$ is the weight matrix of the j-th layer, $\tilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}$ is the adjacency matrix with added self-loops, $\tilde{\mathbf{D}}$ is the diagonal degree matrix $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$, and $\sigma(\cdot)$ denotes a nonlinear activation such as the Rectified Linear Unit (ReLU).

We write the GNN-based graph anomaly detection module as:

$$\hat{\mathbf{y}}, \mathbf{Z} = g_{ad-gnn}(\mathbf{A}, \mathbf{X}; \Theta), \tag{4}$$

where g_{ad-gnn} is a GNN-based model (e.g., DOMINANT [7], GCN [21]), $\mathbf{Z}$ is the latent node representation matrix, and $\hat{\mathbf{y}}$ is the predicted user suspiciousness.

4.2 Action Sequence Augmentation Module

The action sequence augmentation module takes user action sequences as input and outputs the predicted next items that each user is likely to adopt in the future. These predicted actions are then added back to the "user-adopts-item" graph and forms the augmented graph. The main component of this module is a Seq2Seq encoder-decoder network [40], which is robust to model choice and can be any state-of-the-art Seq2Seq model (e.g., GRU [4], Transformer [43], LSTM [19]).

The action sequence augmentation module is designed to capture behavior patterns from the sequential data (i.e., item adoption history) and use them for graph augmentation. In general, it takes the following functional form:

$$\mathbf{A}' = f_{aug}(\mathbf{A}, \mathbf{X}, \hat{\mathbf{y}}; \Theta), \tag{5}$$

where $\mathbf{A}'$ stands for the adjacency matrix with augmented predicted behaviors, $\mathbf{A}$ is the original adjacency matrix, $\mathbf{X}$ is the feature matrix, $\hat{\mathbf{y}}$ is the predicted user suspiciousness by g_{ad} or g_{ad-gnn}, and Θ stands for the trainable parameter.

Sequence prediction. Here we regard each user's action sequence as a sequence of features that are constructed as prior knowledge. For example, for content-based item (e.g., reviews), the features can be the embedded representations of the texts. For each user u's action sequence $x^{(u)} = \{\mathbf{x}_1^{(u)}, \ldots, \mathbf{x}_{l_u}^{(u)}\}$, if a GNN-based anomaly detection module that learns node representation is used, say $\mathbf{z}_u$ for user u, then the action sequence is represented as $x^{(u)} = \{\mathbf{x}_1^{(u)} \oplus \mathbf{z}_u, \ldots, \mathbf{x}_{l_u}^{(u)} \oplus \mathbf{z}_u\}$, where $\oplus$ stands for vector concatenation.

We opt for simplicity and adopt a GRU model to capture contextualized representations of each action in the sequence and make predictions. Hence, the hidden state of each action by user u is:

$$\mathbf{h}_i^{(u)} = \overrightarrow{\text{GRU}}(\mathbf{x}_i^{(u)}, \mathbf{h}_{i-1}^{(u)}), \tag{6}$$

where $\mathbf{h}_i$ refers to the hidden state in the i-th step. Moreover, we use a linear readout function to predict of the next action by

$$\hat{\mathbf{x}}_{i+1}^{(u)} = \mathbf{W}_p \cdot \mathbf{h}_i^{(u)} + \mathbf{b}_p, \tag{7}$$

where $\mathbf{W}_p \in \mathbb{R}^{|\mathbf{h}| \times k}$, $\mathbf{b}_p \in \mathbb{R}^k$ are trainable parameters and $\hat{\mathbf{x}}_{i+1}^{(u)}$ is the predicted feature vector for user u's next action. The decoder can also predict multiple sequential actions.

Sequence Augmentation. Finally, we augment the graph by explicitly predicting the future items for users and adding the predicted behaviors as edges into the graph. When $\hat{\mathbf{x}}_{i+1}^{(u)}$ is predicted by Eq.(7), the next input item is determined by cosine similarity:

$$\mathbf{x}_{i+1}^{(u)} = \underset{\mathbf{x} \in \mathbf{X}^v}{\text{argmax}} \left(\frac{\hat{\mathbf{x}}_{i+1}^{(u)} \cdot \mathbf{x}}{||\hat{\mathbf{x}}_{i+1}^{(u)}|| \times ||\mathbf{x}||} \right). \tag{8}$$

where $\mathbf{X}^v$ is the item feature matrix serving as the vocabulary. Let $v_i^{(u)}$ be the corresponding item to feature $\mathbf{x}_i^{(u)}$. For each user u, the decoder predicts the next Δl_u future actions $\{v_{l_u+1}^{(u)}, ..., v_{l_u+\Delta l_u}^{(u)}\}$. Thereby, the augmented adjacency matrix can be calculated via

$$\mathbf{A}' = \mathbf{A} + \sum_{u \in \mathcal{U}} \sum_{i=l_u+1}^{l_u+\Delta l_u} \mathbf{O}(u, v_i^{(u)}), \tag{9}$$

where $\mathbf{O}$ is an empty matrix except $O_{i,j} = 1$.

We next discuss two strategies for training ELAND.

Algorithm 1: ELAND-ITR

Input : Adjacency matrix $\mathbf{A}$; node feature matrix $\mathbf{X}$; number of iterations I; anomaly detection module $g \in \{g_{ad}, g_{ad-gnn}\}$; action sequence augmentation module f_{aug}.

Output : User prediction results $\hat{\mathbf{y}}$.

```
X_orig = X ;
if g is g_ad-gnn then
    ŷ, Z = g(A, X) ;                    // Defined in Eq.(4)
    X = concat(X_orig, Z) ;
else
    ŷ = g(A, X) ;                       // Defined in Eq.(2)
end
for i in range(I) do
    Re-initialize the parameters Θ in f_aug ;
    A' = f_aug(A, X, ŷ) ;               // Defined in Eq.(5)
    if g is g_ad-gnn then
        Re-initialize the parameters Θ in g ;
        ŷ, Z = g(A', X) ;
        X = concat(X_orig, Z) ;
    else
        Re-initialize the parameters Θ in g ;
        ŷ = g(A', X) ;
    end
end
return ŷ ;
```

4.3 ELAND-ITR: an Iterative Approach

The graph anomaly detection module benefits from the enriched graph structure generated by the action sequence augmentation, and the augmentation module benefits from the detection module. Both modules are interdependent and mutually enhance each other. Hence, we can use a bootstrapping training strategy to iteratively train both modules and jointly optimize their performances.

Algorithm 1 shows the process of ELAND-ITR. We start with the anomaly detection module on the original graph. If the anomaly detection module is GNN-based, the features are updated by concatenating the learned node representations with the original node

features. Then each iteration proceeds as follows: (1) we train a new action sequence augmentation module f_{aug} with the graph and the results given by the detection module g_{ad} or g_{ad-gnn}, (2) we train and make inference with a new initialized graph anomaly detection module on the updated graph structure $\mathbf{A}'$. After multiple iterations, the final prediction result $\hat{\mathbf{y}}$ is reported.

During the inference stage, we utilize the predicted suspiciousness scores $\hat{y}_u$ for each user given by the anomaly detection module to decide the number of predictions we make for user u:

$$\Delta l_u = \lfloor \kappa \cdot \hat{y}_u \rfloor, \tag{10}$$

where $\kappa \in \mathbb{Z}^+$ is a hyperparameter to control the maximum number of predictions. The intuition is that anomalous users tend to perform more actions to achieve their goal (e.g., fake trending topic boosting), so we generate more predicted items for the users that are more likely to be anomalies, via the anomaly detection module.

Training ELAND-ITR. If g_{ad} is unsupervised, it is trained with its own objectives. If supervised, the graph anomaly detection module is trained with the standard binary cross entropy:

$$\mathcal{L}_{ad} = -\sum_{u \in \mathcal{U}} \big(y_u \log(\hat{y}_u) + (1 - y_u) \log(1 - \hat{y}_u)\big). \tag{11}$$

The action sequence augmentation module f_{aug} is trained with the following loss function which maximizes the cosine similarity between the predicted actions and the actual actions.

$$\mathcal{L}_{aug} = -\frac{1}{|\mathcal{U}|} \sum_{u \in \mathcal{U}} \frac{1}{l_u} \sum_{i=1}^{l_u} \frac{\hat{\mathbf{x}}_i^{(u)} \cdot \mathbf{x}_i^{(u)}}{||\mathbf{x}_i^{(u)}|| \times ||\mathbf{x}_i^{(u)}||}. \tag{12}$$

During the training of ELAND-ITR, the two modules (g and f) are trained with the corresponding loss independently and iteratively.

4.4 ELAND-E2E: an End-to-End Approach

In addition to ELAND-ITR, we propose an end-to-end model ELAND-E2E that does not require the iterative training process. Hence, it avoids the potential error propagation issue in bootstrapping.

The anomaly detection module in ELAND-E2E is a neural model that allows training with back-propagation in order to be trained together with the rest of the model. Thus, we use GNN-based models (e.g. DOMINANT [7]) for the detection module. As ELAND-E2E is end-to-end and trained as a whole and the action sequence augmentation module is executed prior to the graph anomaly detection module, the augmentation module should no longer require $\mathbf{y}'$ as input anymore. The module can then be defined as

$$\mathbf{A}' = f_{aug-e2e}(\mathbf{A}, \mathbf{X}; \Theta), \tag{13}$$

When augmenting the graph with $f_{aug-e2e}$, instead of discretely predicting the next items as in Eq. (8), we predict the following items via sampling. For each prediction of each user u, we use the cosine similarity of the predictions $\pi_u = [\pi_{u,1}, \ldots, \pi_{u,n}]$ and apply the Gumbel-Softmax with reparameterization trick [13, 30] to sample π'_u from π_u by:

$$\pi'_u = \frac{\exp\left(\left(\log\left(\pi_u\right) + g_u\right)/\tau\right)}{\sum_{j=1}^{n} \exp\left(\left(\log\left(\pi_{u,j}\right) + g_j\right)/\tau\right)}, \tag{14}$$

where $g_u \sim \text{Gumbel}(0, 1)$ is a random variate sampled from the Gumbel distribution and τ is a temperature hyperparameter controlling the distribution of the results. Smaller τ results in more difference between classes. We then discretize π'_u into one-hot vectors and add it into the adjacency matrix without damaging sparsity.

Algorithm 2: ELAND-E2E

Input : Adjacency matrix $\mathbf{A}$; node feature matrix $\mathbf{X}$; action sequence augmentation module $f_{aug-e2e}$; GNN-based anomaly detection module g_{ad-gnn}; number of training epochs *n_epochs*.
Output : User prediction results $\hat{\mathbf{y}}$.

```
/* model training                                   */
```
Initialize $\Theta_{aug-e2e}$ in $f_{aug-e2e}$ and Θ_{ad-gnn} in g_{ad-gnn} ;
for *epoch in range(n_epochs)* **do**
  $\mathbf{A}' = f_{aug-e2e}(\mathbf{A}, \mathbf{X})$; // Defined in Eq.(13)
  $\hat{\mathbf{y}}, \mathbf{Z} = g_{ad-gnn}(\mathbf{A}', \mathbf{X})$; // Defined in Eq.(4)
  Calculate $\mathcal{L}_{e2e}$ with Eq.(16);
  Update $\Theta_{aug-e2e}$ and Θ_{ad-gnn} with $\mathcal{L}_{e2e}$;
end
```
/* model inferencing                                */
```
$\mathbf{A}' = f_{aug-e2e}(\mathbf{A}, \mathbf{X})$;
$\hat{\mathbf{y}}, \mathbf{Z} = g_{ad-gnn}(\mathbf{A}', \mathbf{X})$;
return $\hat{\mathbf{y}}$;

As $f_{aug-e2e}$ no longer takes $\mathbf{y}'$ as input, we use preferential attachment to calculate the number of predictions for ELAND-E2E. Zang et al. [47] showed that the dynamics of a random variable x of exponential distribution $\frac{dx(t)}{dt}$ is proportional to $x(t)$ (i.e. $\frac{dx(t)}{dt} \propto x(t)$). Since user behavior patterns are generally considered to be consistent, it is reasonable to assume that the item selection process follows a Poisson process, whose frequency distribution could be further generalized to an exponential distribution [5]. Therefore, the number of predictions for user u's actions can be calculated using preferential attachment:

$$\Delta l_u = \left\lfloor \gamma \cdot \frac{d_u}{\sum d} \right\rfloor, \tag{15}$$

where d_u is u's degree, $\sum d$ is the sum of all users' degree, and γ is a hyperparameter that controls the total number of actions to augment the graph. Δl_u actions will be predicted for u, taking the current distribution of actions into consideration.

Training ELAND-E2E. Depending on the graph anomaly detection module, its loss function $\mathcal{L}_{ad}$ can either be custom tailored (unsupervised) or a standard binary cross entropy loss (supervised) as defined in Eq.(11). A cosine similarity loss $\mathcal{L}_{aug}$ as defined in Eq.(12) is still used to train the augmentation module. Thus ELAND-E2E is trained with a multi-task loss, defined as

$$\mathcal{L}_{e2e} = \mathcal{L}_{ad} + \mathcal{L}_{aug}. \tag{16}$$

Algorithm 2 shows the process of ELAND-E2E. In each epoch, the graph structure is augmented by the output from the action sequence augmentation module with the Gumbel-Softmax trick. We use the Straight-Through gradient estimator [13], passing gradients directly through un-discretized probabilities to train. The GNN-based anomaly detection module uses the augmented graph to predict labels $\mathbf{y}'$. The multi-task loss $\mathcal{L}_{e2e}$ defined by Eq.(16) is used to supervise the whole framework.

Table 1: Statistics of the datasets.

	Weibo	Amazon	Reddit
# Users	40,235	3,024	6,000
# Items	5,284	9,355	2,943
# Edges	55,624	2,434,019	79,210
# Actions	75,285	15,822,365	604,919
% Anomaly users	8.2%	20.2%	13.9%

5 EXPERIMENTS

In this section, we evaluate the performance of proposed ELAND-ITR and ELAND-E2E. Our implementation is made publicly available[2].

5.1 Experimental Setup

5.1.1 Datasets.

We evaluate with three real-world datasets across different domains.

Weibo is a micro-blogging dataset [20]. We build "user-(re)posted-microblog" graph where the user nodes are registered users and item nodes are micro-blog posts. An action between a users and an item indicates that the user posted the item. The original dataset [20, 50] contains public user profiles, (re)posting of micro-blogs, and text of all micro-blogs. As the dataset is a micro-blogging graph, we define the anomalous users in this dataset as the social-spam users accounts which continuously post advertisements or malicious links [50]. Due to the large scale (40K+ users), the dataset does not have manually annotated golden labels. So we labelled the users by their post text, profile information, and their behavior time following previous works [20, 50]. Specifically, we use the following criteria to label anomalies: (1) *Social spambot accounts:* The major conspicuous characteristic of suspicious users is their bot-controlled behavior. As we observed, most suspicious users posted in a fixed set of time intervals. For example, a user is identified as an anomaly if more than 2/3 of the time intervals of his posts are within 30 ± 2 seconds. (2) *Accounts with suspicious posts:* We checked the post text and spotted the accounts whose posts are mostly advertisements or content with malicious links. We follow the train/validation/test split in previous works [50] where we randomly pick 5000 users as training set, 5000 users as validation set, and the rest users as testing set.

Amazon Reviews is a review dataset from Amazon [32]. We build "user-used-word" graph where user nodes are amazon users and item nodes are words with polarity bias from Hedonometer's word list [10]. An action between a user and an item indicates that the user used that word in a review. The original dataset [32] contains reviews on Amazon under the directory of Video Games. Following previous works [23, 49], ground truth is is defined using helpfulness votes, which is indicative of malicious [23] behavior. Users with at least 50 votes are labeled benign if the fraction of helpful-to-total votes is ≥ 0.75, and fraudulent if ≤ 0.25. We randomly split the users in to train/validation/test sets with the ratio of 20%/20%/60%.

Reddit is a forum dataset [2]. We build "user-commented-subreddit" graph where user nodes are reddit users and item nodes are subreddits. An action between a user and an item indicates that the user commented under that subreddit. Due to the huge size of the original dataset [2] which contains entire public Reddit comments since 2005, we used the part of data from September 2019. As the dataset gives an score for each comment (# of up votes minus # of down votes), we label the users according similar rules for labeling of the Amazon dataset [23, 49]. We selected users who received at least 10 scores with absolute value ≥ 10 (indicating at least 100 votes). Users are benign if all scores he/she received are positive; users are anomaly if $\geq 75\%$ of the scores he/she receive are ≤ -10. We randomly split the users in to train/validation/test sets with the ratio of 20%/20%/60%.

For all datasets, more actions between the same user and item would result in an edge with higher weight. Statistics for the datasets are shown in Table 1.

5.1.2 Baselines.

We evaluate ELAND with the following methods as g_{ad} module:

- GCN [21]: A supervised graph neural network. The neural structure has been empirically demonstrated to be useful in the area of graph anomaly detection [49].
- GRAPHSAGE [18]: An inductive graph neural network that can also be used for supervised node classification.
- HETGNN [48]: A supervised graph neural network that handles the heterogeneous graphs of multiple types of nodes.
- DOMINANT [7]: An unsupervised graph anomaly detection method designed based on graph auto-encoder.
- DEEPAE [55]: An unsupervised graph anomaly detection method with multi-order proximity preservation.

Moreover, we compare ELAND with the following baseline methods:

- RNNFD [3]: A RNN-based model for fraud detection on sequential data in industry applications.
- GRAND [15]: A GNN-based model that leverages graph data augmentation to regularize the optimization process.
- JODIE [24]: A bipartite graph interaction prediction method by dynamic embedding trajectory learning.
- GAUG [53]: A graph data augmentation method designed for semi-supervised node classification with GNNs.

As JODIE [24] and GAUG [53] can be considered as model-agnostic graph data augmentation methods, we also report their performance across different g_{ad} modules.

5.1.3 Implementation Details.

All experiments were conducted on Linux servers with Intel Xeon Gold 6130 Processor (16 Cores @2.1Ghz), 96 GB of RAM, and 4 NVIDIA Tesla V100 cards (32 GB of RAM each) or RTX 2080Ti cards (11 GB of RAM each).

For all methods, we used hidden dimension of 128 and Adam optimizer. All methods have weight decay of 5e−4. For GRAPHSAGE, we use the mean aggregator. To make fair comparisons, these aforementioned parameters are fixed for all experiments. For GAUG [53], we used the variant of GAUGM because GAUGO gets CUDA out of memory error on our datasets. We used the official implementation from the authors for GRAND [15], JODIE [24], and GAUG [53]. We report the average and standard deviation of all performances in 20 runs with random parameter initialization.

ELAND-ITR: κ is selected w.r.t. the average length of action sequences in each dataset. For Weibo and Reddit datasets, $\kappa = 150$; for Amazon Reviews dataset, $\kappa = 5000$.

ELAND-E2E: As ELAND-E2E is more robust to γ, we opt for simplicity and use $\gamma = 100$ for all datasets. During the training of

[2] https://github.com/DM2-ND/Eland

Table 2: ELAND performance across supervised graph anomaly detection methods with only earliest 10% of data available.

Anomaly detection module g_{ad}	Method	Weibo AUC	Weibo AP	Amazon Reviews AUC	Amazon Reviews AP	Reddit AUC	Reddit AP
	RNNFD [3]	54.52±0.12	17.44±0.10	60.22±0.29	28.61±0.11	66.08±0.36	26.45±0.83
	GRAND [15]	82.58±2.11	40.12±2.99	81.71±2.56	57.66±2.98	79.09±0.18	42.37±0.72
GCN [21]	Original	81.78±0.78	41.21±1.36	80.28±0.09	57.73±0.21	78.01±0.71	41.21±0.69
	+JODIE [24]	67.80±1.30	17.12±2.72	–	–	73.12±2.13	31.62±3.98
	+GAUG [53]	82.04±0.40	48.17±0.59	81.91±0.02	60.12±0.15	78.78±0.07	40.74±0.72
	+ELAND-ITR	82.76±0.71	48.51±1.06	80.85±0.67	58.14±0.39	78.94±0.83	43.11±1.22
	+ELAND-E2E	**84.14**±0.50	**54.15**±0.83	**85.54**±0.46	**65.48**±0.14	**79.58**±0.38	**44.60**±0.43
GRAPHSAGE [18]	Original	81.87±0.56	45.26±2.54	78.67±0.09	58.00±0.07	81.06±0.02	47.71±0.01
	+JODIE [24]	69.44±0.95	16.01±2.09	–	–	74.66±0.09	34.70±0.06
	+GAUG [53]	82.10±0.46	47.81±1.29	80.79±0.02	56.38±0.03	81.37±0.01	43.83±0.01
	+ELAND-ITR	82.34±0.50	48.40±0.91	**81.59**±0.23	**59.91**±0.12	**81.62**±0.10	**48.25**±0.11
	+ELAND-E2E	**83.41**±0.37	**50.61**±0.93	79.92±0.19	58.21±0.31	79.83±0.02	44.38±0.02
HETGNN [48]	Original	81.33±0.43	39.66±1.48	86.24±0.13	67.98±0.25	91.51±0.13	67.51±0.17
	+JODIE [24]	68.99±0.44	17.38±1.87	–	–	92.02±0.36	68.16±0.30
	+GAUG [53]	82.09±0.21	47.05±0.51	87.26±0.12	71.76±0.33	91.99±0.02	66.30±0.25
	+ELAND-ITR	81.46±0.57	40.20±1.19	**90.58**±0.86	**75.08**±0.57	**92.44**±0.07	**69.31**±0.29
	+ELAND-E2E	**84.09**±0.55	**54.07**±1.64	87.57±0.26	68.46±0.35	84.24±0.22	55.34±0.88

Table 3: ELAND performance across unsupervised graph anomaly detection methods with only earliest 10% of data available. DOMINANT and DEEPAE are not valid on Amazon data due to their design heuristic, hence their results are not included.

Anomaly detection module g_{ad}	Method	Weibo AUC	Weibo AP	Reddit AUC	Reddit AP
DOMINANT [7] (Unsupervised)	Original	56.77±1.96	13.73±1.22	61.23±0.35	18.30±0.21
	+JODIE [24]	58.18±0.77	11.09±0.13	61.64±0.09	18.81±0.06
	+GAUG [53]	61.22±1.86	14.15±2.38	62.26±2.70	17.09±1.39
	+ELAND-ITR	**65.44**±1.78	19.42±1.29	**62.96**±0.10	18.90±0.04
	+ELAND-E2E	63.91±0.92	**21.90**±0.87	61.73±0.27	**18.91**±0.14
DEEPAE [55] (Unsupervised)	Original	56.10±2.01	12.65±1.31	61.94±0.39	18.29±0.13
	+JODIE [24]	57.74±0.87	11.16±0.73	61.57±0.32	18.93±0.06
	+GAUG [53]	61.18±2.03	11.58±1.27	61.29±0.82	18.23±0.53
	+ELAND-ITR	**63.34**±0.82	15.88±0.73	**62.87**±0.37	**19.02**±0.11
	+ELAND-E2E	62.80±3.60	**16.99**±3.87	62.47±0.11	18.88±0.04

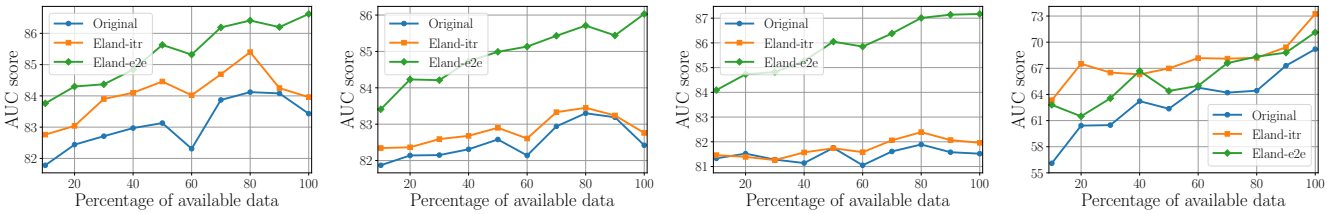
ELAND-E2E, we linearly anneal the temperature of Gumbel-softmax distribution throughout the all training iterations, from $\tau = 5$ (a very flat distribution) to $\tau = 0.5$ (a very peaked distribution).

5.2 Experimental Results

Table 2 and 3 report the performance of our proposed ELAND and baseline methods over supervised learning and unsupervised learning methods, respectively. These tables are organized per anomaly detection algorithm (row), per dataset (column), and per augmentation method (within-row). We report AUC and Average Precision (AP). Note that results of JODIE [24] on Amazon are missing due to CUDA out of memory when running the code from authors on V100 GPU with 32GB RAM.

5.2.1 Enhancing graph anomaly detection methods.
Table 2 shows that ELAND achieves improvements over baseline supervised graph anomaly detection methods, with the only exception of ELAND-E2E with GRAPHSAGE and HETGNN on Reddit. Specifically, ELAND-ITR improves (averaged across datasets) 1.0% (GCN), 1.7% (GRAPHSAGE), and 2.1% (HETGNN) on AUC score and 7.7%, 3.8%, and 4.8% on AP; ELAND-E2E improves 3.8%, 0.7%, and 3.1% on AUC and 17.7%, 1.7%, and 6.3% on AP, respectively.

We also observe that ELAND outperforms all four baseline methods. For RNNFD, its results are not as good as any other methods as it only used the action sequence data. For GRAND, our proposed ELAND-ITR and ELAND-E2E achieve 3.1% and 4.1% improvements on AUC, respectively. JODIE hurts the performances of the original GNNs possibly because it was designed for predicting the next interaction item but not the following action sequence. Hence ELAND outperforms JODIE with large margins. On average, ELAND-ITR and ELAND-E2E improve 12.7% and 14.7% on AUC, respectively. Finally, ELAND also outperforms the graph data augmentation method GAUG. On average, ELAND-ITR and ELAND-E2E improve 5.4% and 1.1% on AUC, respectively. We observe that ELAND-E2E tend to perform better for supervised graph anomaly detection methods.



(a) GCN [21] (b) GraphSAGE [18] (c) HetGNN [48] (d) DeepAE [55]

Figure 3: Eland-itr and Eland-e2e outperform the original version of (a)-(c) supervised graph learning methods and (d) unsupervised graph anomaly detection method in terms of AUC with different amount of available training data.

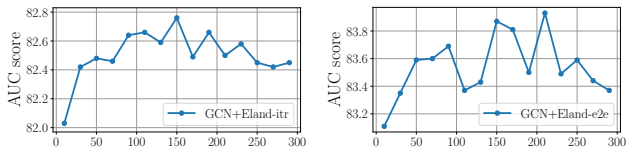
Table 4: Eland is robust to the choice of Seq2Seq model used in the action sequence augmentation module.

Metric	Method	Weibo 10%	Weibo 20%
AUC	GCN	81.78±0.78	82.44±0.54
	+Eland-e2e (LSTM)	83.76±0.74	84.30±0.53
	+Eland-e2e (RNN)	83.83±0.34	84.33±0.30
	+Eland-e2e (GRU)	84.14±0.50	84.57±0.36
AP	GCN	41.21±1.36	44.64±3.25
	+Eland-e2e (LSTM)	51.31±2.42	51.46±1.12
	+Eland-e2e (RNN)	52.74±0.63	52.84±0.92
	+Eland-e2e (GRU)	54.15±0.83	55.14±0.66

From Table 3 we observe that Eland outperforms unsupervised graph anomaly detection baselines. Specifically, Eland-itr improves by 9.0% (Dominant), and 7.2% (DeepAE) on AUC and 22.4%, and 14.8% on AP; Eland-e2e improves 6.7%, and 6.4% on AUC and 31.4%, and 18.8% on AP, respectively. Eland also outperforms both alternative methods JODIE and GAug. On average, Eland-itr improves 5.1% on AUC and 25.9% on AP; Eland-e2e improves 3.6% on AUC and 33.2% on AP. We note that Eland-itr achieves better performances than Eland-e2e with unsupervised anomaly detection modules. This is due to the misalignment of the training objectives of unsupervised methods and the evaluation metrics, resulting in the unsupervised objectives that mislead the action sequence augmentation module during end-to-end training.

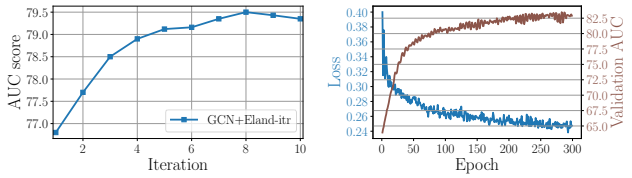
We find that on the Amazon dataset, Dominant and DeepAE are not effective, so their results are not included. The reason is that the unsupervised training objectives in the two methods are based on outlier detection with auto-encoders, which is not aligned with the actual label distributions. Therefore, most users that were marked as "highly suspicious" by these two methods in the Amazon data are actually benign users.

5.2.2 Achieving early anomaly detection.
To better show the results of the proposed Eland framework for early anomaly detection, we present Figure 3 (along with Figure 1 for Dominant), in which we show the trends of AUC scores of original graph anomaly detection methods and our proposed framework change as more observed data is available on the Weibo dataset. We observe that in general, both Eland-itr and Eland-e2e are able to accomplish the early-stage anomaly detection task as defined in Section 3. Eland-e2e generally achieves more improvements on supervised graph anomaly detection methods; for unsupervised methods, Eland-itr tends to perform better, which aligns with our observations from Table 2 and 3.



(a) κ (b) γ

Figure 4: Eland-itr and Eland-e2e are robust to the hyper-parameters κ and γ.



(a) Eland-itr with GCN (b) Eland-e2e with GCN

Figure 5: Convergence process of the proposed Eland-itr and Eland-e2e. When it came to the 8th iteration (in Eland-itr) or about 200 epochs (in Eland-e2e), the AUC of the two methods converge.

Particularly, for supervised methods, the performance of Eland-e2e with only 10% or 20% available data is better or comparable with the baselines with all data. With only 20% (GCN) / 10% (GraphSAGE) / 10% (HetGNN) / 40% (Dominant) / 60% (DeepAE) of earliest available data, Eland achieves better or comparable performance than the original model with all available data.

Although the original performances are not monotonically increasing (though correlated), it is also worth mentioning that for all of the baseline detection methods, we are able to observe substantial improvements in absolute performance on average, which are evidence for Eland's powerful ability to model the evolution of the graph structure and user-item interactions.

5.2.3 Ablation study and sensitivity analysis.
Ablation study on Seq2Seq choice. To validate the robustness of

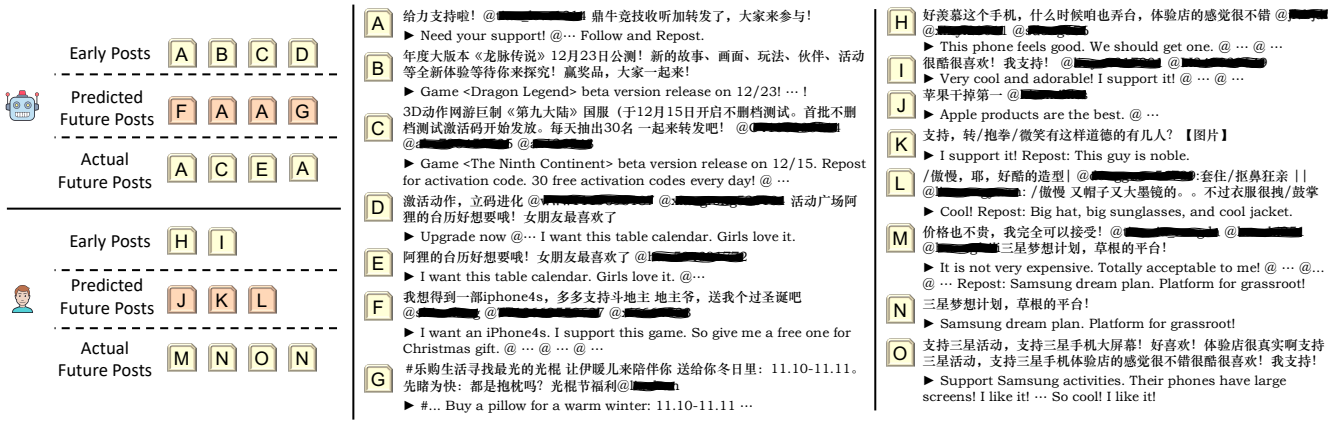


Figure 6: Case study: the historical posts, predicted posts (by ELAND-E2E), actual future posts sequences of two users, and actual post content each followed by a brief translation (marked with solid triangles). This study shows the previously wrong classified users can be correctly classified with the predicted posts by ELAND-E2E.

ELAND on the choice of Seq2Seq model in the action sequence augmentation module, we show the results of ELAND-E2E with different Seq2Seq models (GRU [4], LSTM [19], and traditional RNN [19]) on Weibo dataset in Table 4. We can observe that GCN+ELAND-E2E with all Seq2Seq methods can outperform vanilla GCN [21], indicating that ELAND is robust to the choice of Seq2Seq method used in the action sequence augmentation module. Future works can try more powerful sequential models such as Transformer [43] as the g_{aug} module.

Hyperparameter sensitivity. We test through ELAND's hyperparameters κ and γ, which controls the amount of augmented actions for each user in ELAND-ITR and ELAND-E2E, respectively. Figure 4 shows that the proposed ELAND-ITR and ELAND-E2E are robust to κ and γ for range of $30 \leq \{\kappa, \gamma\} \leq 290$ for Weibo data.

Model convergence. Figure 5 presents the convergence progress of ELAND. Figure 5a shows that the change of performance over iterations of ELAND-ITR with GCN. We observe a smooth yet fast increase of the AUC at the first few iterations, which then reaches a steady state. The curve shows the bootstrapping iterative training design of ELAND-ITR is meaningful and demonstrates the mutual beneficial relationship between the two modules. Figure 5b shows the convergence of loss and validation AUC during training.

5.3 Case Study

To further demonstrate the effectiveness of our proposed ELAND framework, we conduct case studies on the Weibo data. We study two user cases: (1) an anomalous user who was falsely classified as a benign user at early stage when data were not complete, and (2) a benign user who was falsely classified as an anomaly at an early stage. Figure 6 presents their historical posts, predicted posts (given by ELAND-E2E), and actual future posts.

For the anomalous user, GRAPHSAGE originally classified him as a benign user with confidence of 0.73. ELAND-E2E successfully classified him as an anomaly with confidence of 0.58. From the posts of this user we observe that the user was posting advertisements in early posts. However, the posts did not repeat or show any strong pattern of an anomaly. ELAND-E2E correctly predicted that he will repeat posting the message *A* and hence enforced his suspicious pattern. The ground truth showed that this user actually did continue to repeat the advertisement posts (e.g., repeat posting message *A*). We verified that this user was an anomaly by checking the complete history of his posts and we found that the history has a large number of advertisements.

For the other user, GRAPHSAGE falsely predicted this benign user as an anomaly with confidence of 0.73. With ELAND-E2E, this user was correctly classified as a benign user with confidence of 0.74. One possible reason is that this user mentioned cell phones in his early posts and there were a large number of advertisement posts about phones in the dataset. ELAND-E2E predicted that he would be posting some unrelated posts, one of which was related with iPhone. Although the predicted future posts were not exactly the same as the actual future posts, the predicted posts showed that the behavior pattern of this user is dissimilar to that of an anomalous user. We verified that this user was benign. Most of his posts were about personal ideas. His posts that supported Samsung phones caused the false positive classified by GRAPHSAGE.

6 CONCLUSIONS

In this work, we proposed ELAND for early graph-based anomaly detection via action sequence augmentation. ELAND aimed at improving existing graph anomaly detection methods with limited observations. Our work managed to model both the node representation and graph topology evolution through behavior forecasting via action sequence augmentation. Experiments on real-world data demonstrated that ELAND improves graph anomaly detection methods towards early and accurate anomaly detection.

ACKNOWLEDGMENTS

This research was supported in part by Snap Research Fellowship and National Science Foundation (NSF) Grants no. IIS-1849816 and no. CCF-1901059.

REFERENCES

[1] Leman Akoglu, Hanghang Tong, and Danai Koutra. 2015. Graph based anomaly detection and description: a survey. *Data mining and knowledge discovery* (2015), 626–688.

[2] Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. 2020. The pushshift reddit dataset. In *ICWSM*, Vol. 14. 830–839.

[3] Bernardo Branco, Pedro Abreu, Ana Sofia Gomes, Mariana SC Almeida, João Tiago Ascensão, and Pedro Bizarro. 2020. Interleaved Sequence RNNs for Fraud Detection. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*.

[4] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[5] John CB Cooper. 2005. The poisson and exponential distributions. *Mathematical Spectrum* 37, 3 (2005), 123–125.

[6] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*. 3844–3852.

[7] Kaize Ding, Jundong Li, Rohit Bhanushali, and Huan Liu. 2019. Deep anomaly detection on attributed networks. In *Proceedings of the 2019 SIAM International Conference on Data Mining*. SIAM, 594–602.

[8] Kaize Ding, Jundong Li, and Huan Liu. 2019. Interactive anomaly detection on attributed networks. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 357–365.

[9] Kaize Ding, Qinghai Zhou, Hanghang Tong, and Huan Liu. 2021. Few-shot Network Anomaly Detection with Cross-network Meta-learning. In *WWW*.

[10] Peter Sheridan Dodds, Eric M Clark, Suma Desu, Morgan R Frank, Andrew J Reagan, Jake Ryland Williams, Lewis Mitchell, Kameron Decker Harris, Isabel M Kloumann, James P Bagrow, et al. 2015. Human language reveals a universal positivity bias. *Proceedings of the national academy of sciences* 112, 8 (2015), 2389–2394.

[11] Manqing Dong, Feng Yuan, Lina Yao, Xiwei Xu, and Liming Zhu. 2020. MAMO: Memory-Augmented Meta-Optimization for Cold-start Recommendation. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 688–697.

[12] Yingtong Dou, Zhiwei Liu, Li Sun, Yutong Deng, Hao Peng, and Philip S Yu. 2020. Enhancing graph neural network-based fraud detectors against camouflaged fraudsters. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 315–324.

[13] Ben Poole Eric Jang, Shixiang Gu. 2016. Categorical Reparameterization with Gumbel Softmax. In *International Conference on Learning Representations (ICLR) 2017*.

[14] Dhivya Eswaran, Christos Faloutsos, Sudipto Guha, and Nina Mishra. 2018. Spotlight: Detecting anomalies in streaming graphs. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 1378–1386.

[15] Wenzheng Feng, Jie Zhang, Yuxiao Dong, Yu Han, Huanbo Luan, Qian Xu, Qiang Yang, Evgeny Kharlamov, and Jie Tang. 2020. Graph Random Neural Networks for Semi-Supervised Learning on Graphs. *Advances in Neural Information Processing Systems* 33 (2020).

[16] Hongyang Gao, Zhengyang Wang, and Shuiwang Ji. 2018. Large-scale learnable graph convolutional networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1416–1424.

[17] Zhichun Guo, Chuxu Zhang, Wenhao Yu, John Herr, Olaf Wiest, Meng Jiang, and Nitesh V Chawla. 2021. Few-Shot Graph Learning for Molecular Property Prediction. In *Proceedings of the Web Conference 2021*. 2559–2567.

[18] Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. In *Advances in Neural Information Processing Systems*. 1024–1034.

[19] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[20] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2016. Inferring lockstep behavior from connectivity pattern in large graphs. *Knowledge and Information Systems* 48, 2 (2016), 399–428.

[21] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[22] Yu Kong, Zhiqiang Tao, and Yun Fu. 2017. Deep sequential context networks for action prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 1473–1481.

[23] Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and VS Subrahmanian. 2018. Rev2: Fraudulent user prediction in rating platforms. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 333–341.

[24] Srijan Kumar, Xikun Zhang, and Jure Leskovec. 2019. Predicting dynamic embedding trajectory in temporal interaction networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1269–1278.

[25] Yuening Li, Xiao Huang, Jundong Li, Mengnan Du, and Na Zou. 2019. Specae: Spectral autoencoder for anomaly detection in attributed networks. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2233–2236.

[26] Jun Liu, Amir Shahroudy, Gang Wang, Ling-Yu Duan, and Alex C Kot. 2018. SSNet: scale selection network for online 3D action prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 8349–8358.

[27] Siwei Liu, Iadh Ounis, Craig Macdonald, and Zaiqiao Meng. 2020. A Heterogeneous Graph Neural Model for Cold-Start Recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2029–2032.

[28] Yao Ma, Ziyi Guo, Zhaocun Ren, Jiliang Tang, and Dawei Yin. 2020. Streaming graph neural networks. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 719–728.

[29] Yao Ma, Xiaorui Liu, Tong Zhao, Yozen Liu, Jiliang Tang, and Neil Shah. 2020. A unified view on graph neural networks as graph signal denoising. *arXiv preprint arXiv:2010.01777* (2020).

[30] Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. 2017. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *International Conference on Learning Representations*.

[31] Franco Manessi, Alessandro Rozza, and Mario Manzo. 2020. Dynamic graph convolutional networks. *Pattern Recognition* 97 (2020), 107000.

[32] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *Proceedings of the 22nd international conference on World Wide Web*. 897–908.

[33] Guansong Pang, Chunhua Shen, and Anton van den Hengel. 2019. Deep anomaly detection with deviation networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 353–362.

[34] Aldo Pareja, Giacomo Domeniconi, Jie Chen, Tengfei Ma, Toyotaro Suzumura, Hiroki Kanezashi, Tim Kaler, Tao B Schardl, and Charles E Leiserson. 2020. EvolveGCN: Evolving Graph Convolutional Networks for Dynamic Graphs.. In *AAAI*. 5363–5370.

[35] Kiran Rama, Pradeep Kumar, and Bharat Bhasker. 2019. Deep Learning to Address Candidate Generation and Cold Start Challenges in Recommender Systems: A Research Survey. *arXiv preprint arXiv:1907.08674* (2019).

[36] Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 985–994.

[37] Aravind Sankar, Yozen Liu, Jun Yu, and Neil Shah. 2021. Graph Neural Networks for Friend Ranking in Large-scale Social Platforms. In *WWW*.

[38] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. 2018. Structured sequence modeling with graph convolutional recurrent networks. In *International Conference on Neural Information Processing*. Springer, 362–373.

[39] Kai Shu, Guoqing Zheng, Yichuan Li, Subhabrata Mukherjee, Ahmed Hassan Awadallah, Scott Ruston, and Huan Liu. 2020. Leveraging Multi-Source Weak Social Supervision for Early Detection of Fake News. *arXiv:2004.01732*.

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*.

[41] Xianfeng Tang, Yozen Liu, Neil Shah, Xiaolin Shi, Prasenjit Mitra, and Suhang Wang. 2020. Knowing your FATE: Friendship, Action and Temporal Explanations for User Engagement Prediction on Social Apps. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*.

[42] Kai Ming Ting, Bi-Cun Xu, Takashi Washio, and Zhi-Hua Zhou. 2020. Isolation Distributional Kernel: A New Tool for Kernel based Anomaly Detection. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 198–206.

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 6000–6010.

[44] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[45] Daixin Wang, Jianbin Lin, Peng Cui, Quanhui Jia, Zhen Wang, Yanming Fang, Quan Yu, Jun Zhou, Shuang Yang, and Yuan Qi. 2019. A semi-supervised graph attentive network for financial fraud detection. In *2019 IEEE International Conference on Data Mining (ICDM)*. IEEE, 598–607.

[46] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 974–983.

[47] Chengxi Zang, Peng Cui, Wenwu Zhu, and Fei Wang. 2019. Dynamical Origins of Distribution Functions. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 469–478.

[48] Chuxu Zhang, Dongjin Song, Chao Huang, Ananthram Swami, and Nitesh V Chawla. 2019. Heterogeneous graph neural network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*.

[49] Shijie Zhang, Hongzhi Yin, Tong Chen, Quoc Viet Nguyen Hung, Zi Huang, and Lizhen Cui. 2020. GCN-Based User Representation Learning for Unifying Robust Recommendation and Fraudster Detection. *SIGIR (Information retrieval) 20* (2020).

[50] Tong Zhao, Chuchen Deng, Kaifeng Yu, Tianwen Jiang, Daheng Wang, and Meng Jiang. 2020. Error-Bounded Graph Anomaly Loss for GNNs. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 1873–1882.

[51] Tong Zhao, Tianwen Jiang, Neil Shah, and Meng Jiang. 2021. A Synergistic Approach for Graph Anomaly Detection with Pattern Mining and Feature Learning. *IEEE Transactions on Neural Networks and Learning Systems* (2021).

[52] Tong Zhao, Gang Liu, Daheng Wang, Wenhao Yu, and Meng Jiang. 2021. Counterfactual Graph Learning for Link Prediction. *arXiv preprint arXiv:2106.02172* (2021).

[53] Tong Zhao, Yozen Liu, Leonardo Neves, Oliver Woodford, Meng Jiang, and Neil Shah. 2021. Data Augmentation for Graph Neural Networks. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence*.

[54] Tong Zhao, Matthew Malir, and Meng Jiang. 2018. Actionable objective optimization for suspicious behavior detection on large bipartite graphs. In *2018 IEEE International Conference on Big Data (Big Data)*. IEEE, 1248–1257.

[55] Dali Zhu, Yuchen Ma, and Yinlong Liu. 2020. Anomaly Detection with Deep Graph Autoencoders on Attributed Networks. In *2020 IEEE Symposium on Computers and Communications (ISCC)*. IEEE, 1–6.