IEEE TKDE 2020

Graph Neural Network for Fraud Detection via Spatial-temporal Attention

Dawei Cheng, Xiaoyang Wang, Ying Zhang and Liqing Zhang

现有工作

- 基于规则：专家生成规则
- 基于机器学习：提取特征，有监督学习分类

- 对于信用卡欺诈问题存在时空的高度聚集性
- 时间：一旦发现可疑交易，持卡人会尽快冻结信用卡，诈骗者被要求在短时间内达到信用额度
- 空间：诈骗者更频繁地利用少数易受攻击的商家/设备，这导致欺诈交易从少数特定的地点被打击，而不是由普通的持卡人发行的许多不同的地点

- **将Attention使用在时空的信息上**
- **3D卷积在时空特征提取上具有优势**

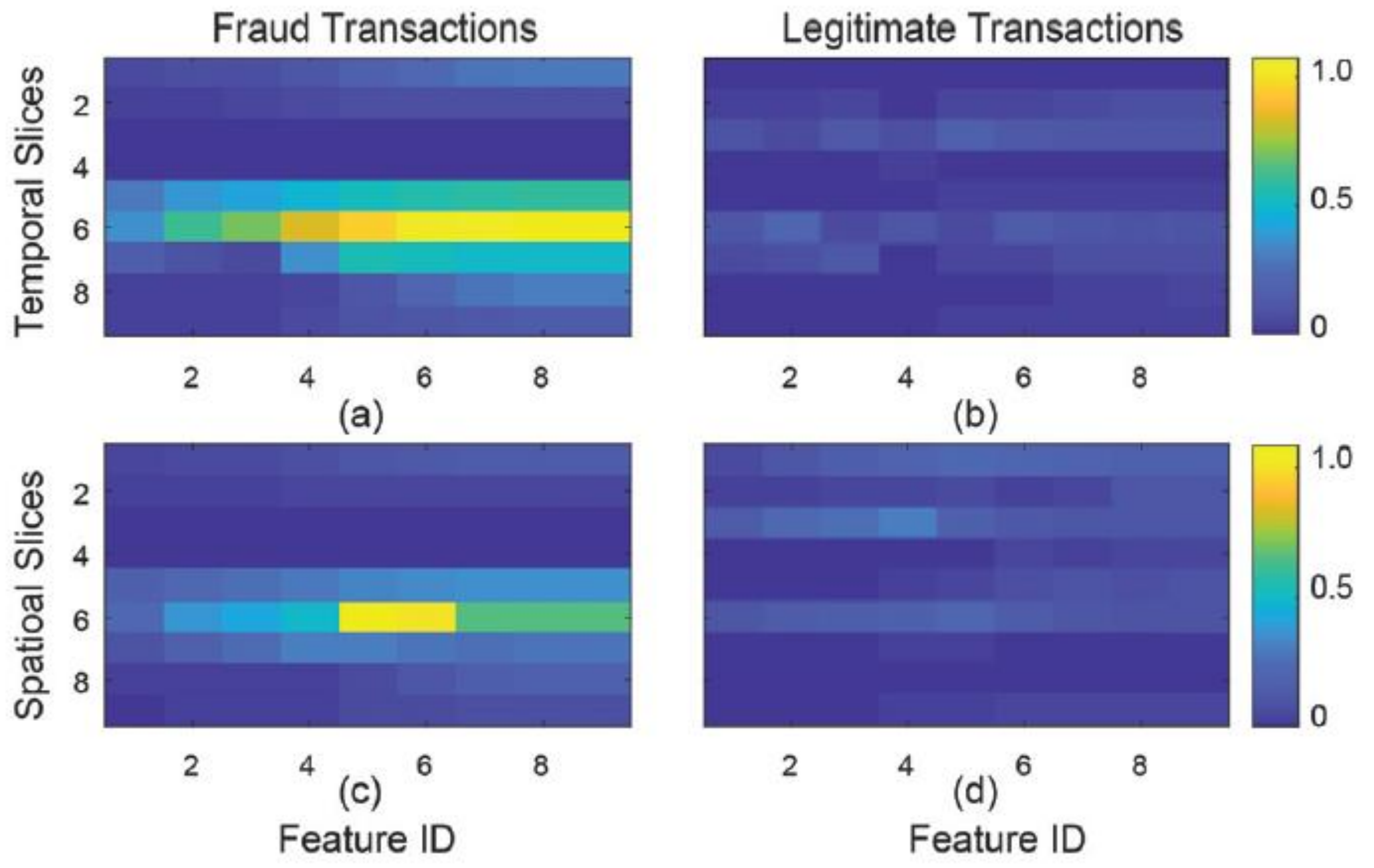


Figure 2: Heat maps of spatio-temporal feature slices from both fraudulent and legitimate transactions.

Model Architecture and Feature Construction

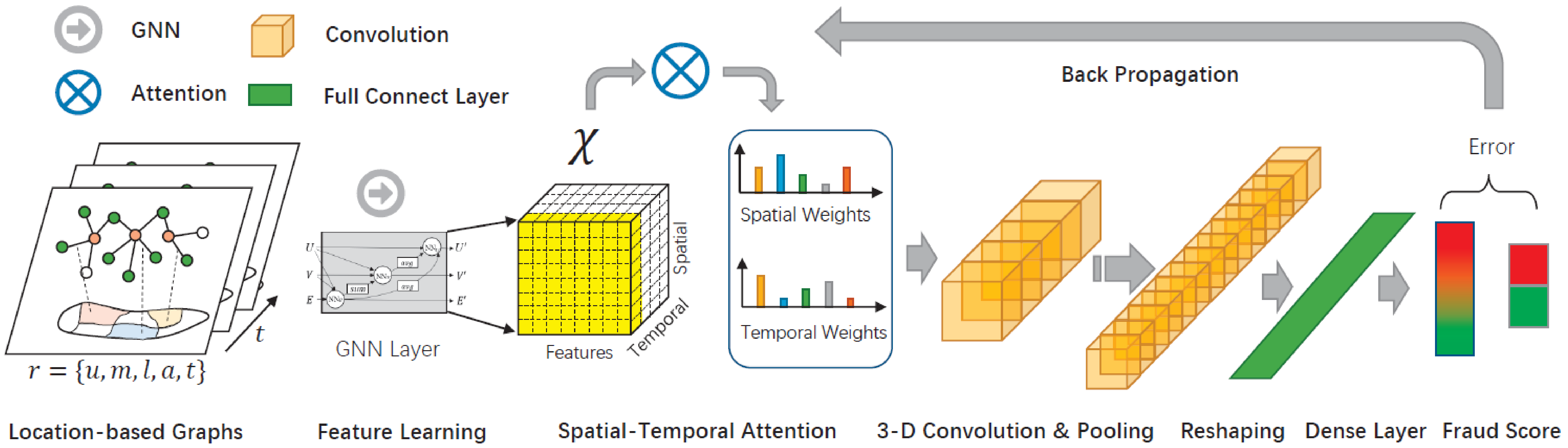


Fig. 3. The illustration of the proposed spatial-temporal attention-based neural network (STAGN) model. Raw transaction records are processed by location-based GNN layer, spatial-temporal attention, and multiple 3D ConvNet to learn high-level representations. Afterward, the learned representations are reshaped to vectors and fed into a detection network for fraud estimation. Attentional weights are jointly optimized in an end-to-end mechanism with 3D convolution and detection networks.

$r = \{u, t, l, m, a\}$,u是用户，t是时间，l是位置编号，m是收款人/机器编号，a是付款数量

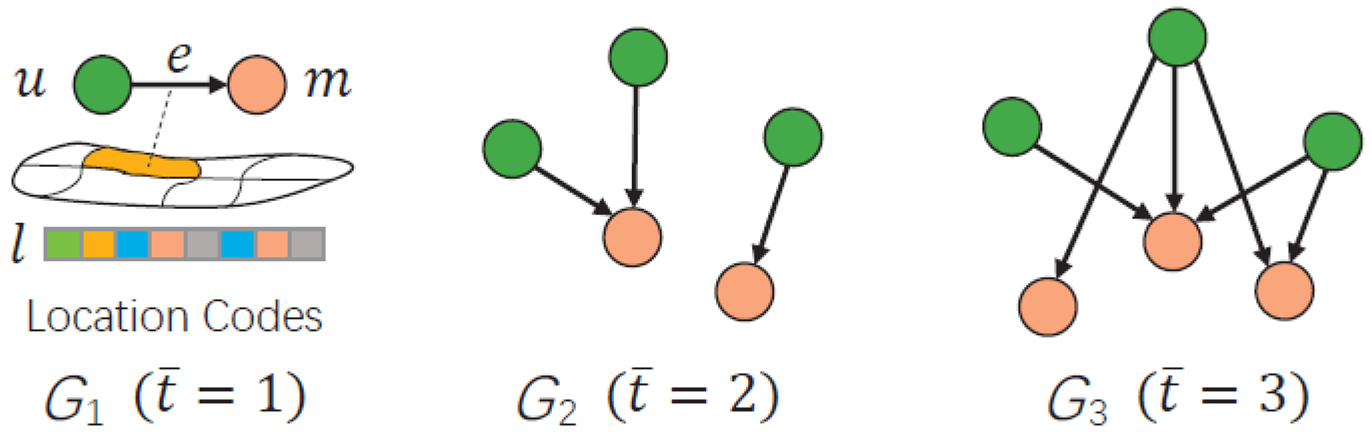
时间切片：不同时间窗口的特征

空间切片，不同空间位置的特征

问题描述：给定一组交易记录 $R = \{U，T，L，M，A\}$ ，一组欺诈事件D，以及时间窗口 $ti \& ti + 1$ ，用于 $(ti，ti + 1]$ 我们要根据从 $t1$ 到 ti 的记录推断是否为欺诈事件，目的是实现高精度的欺诈预测，并探索信用卡交易的欺诈模式

一个r对应一个3-D矩阵，表示了交易特征和基于位置图的特征

v_e 是边信息（交易数、位置），v_g 全局的图特征，v_u, v_m 是随机的特征向量， v_v 是v_u, v_m的结合



Location-based Graph Neural Network

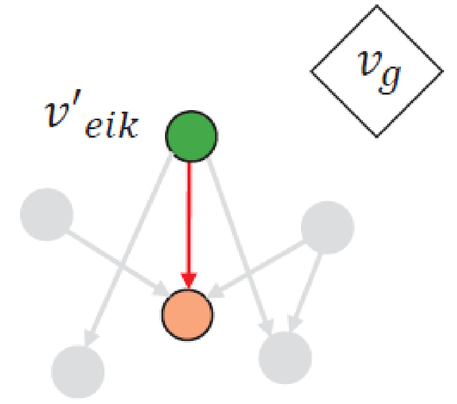
$$v'_{eik} = \mathrm{NN}_e(v_{eik}, I_{ui}, I_{mk}, v_g)$$

$$v'_{ui} = \mathrm{NN}_v\left(\sum_{k=1}^{N_m}(v'_{eik}), v_{ui}, v_g\right)$$

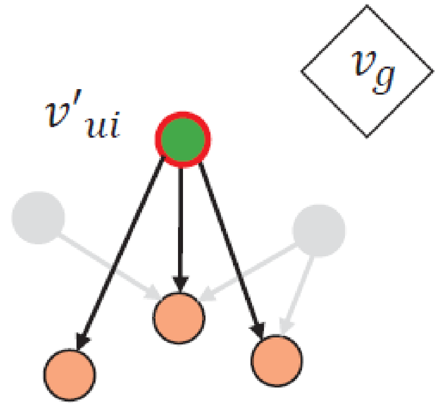
$$v'_{mk} = \mathrm{NN}_v\left(\sum_{i=1}^{N_u}(v'_{eik}), v_{mk}, v_g\right)$$

$$v'_g = \mathrm{NN}_g(\overline{v}_u, \overline{v}_m, \overline{v}_e, v_g)$$

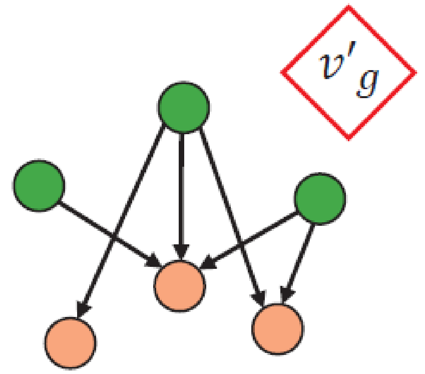
$$\overline{v}_u = \frac{1}{N_u}\sum_i^{N_u} v'_{ui},\ \overline{v}_m = \frac{1}{N_m}\sum_k^{N_m} v'_{mk}$$



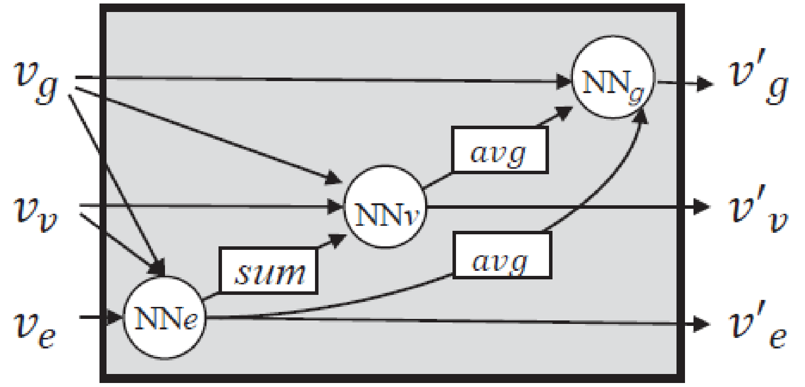
(a) Edge Update



(b) Node Update



(c) Global Update



(d) Full GNN Layer

Spatial-temporal Attention Net

Temporal Attention Layer

$$rept = \sum_{\bar{t}=1}^{N_1} a_{1,\bar{t}} \mathcal{X}(\bar{t}, :, :)$$

$$a_{1,\bar{t}} = \frac{\exp\left((1-\lambda_1) \cdot \mathrm{NN}_{\bar{t}}(W_{\bar{t}}, \mathcal{X}(\bar{t}, :, :))\right)}{\sum_{\bar{t}=1}^{N_1} \exp\left((1-\lambda_1) \cdot \mathrm{NN}_{\bar{t}}(W_{\bar{t}}, \mathcal{X}(\bar{t}, :, :)\right)}$$

Spatial Attention Layer

$$\mathcal{H}^a = \sum_{\bar{s}=1}^{N_2} a_{2,\bar{s}} rept(:, \bar{s}, :)$$

$$a_{2,\bar{s}} = \frac{\exp\left((1-\lambda_2) \cdot \mathrm{NN}_{\bar{s}}(W_{\bar{s}}, rept(:, \bar{s}, :))\right)}{\sum_{\bar{s}=1}^{N_2} \exp\left((1-\lambda_2) \cdot \mathrm{NN}_{\bar{s}}(W_{\bar{s}}, rept(:, \bar{s}, :))\right)}$$

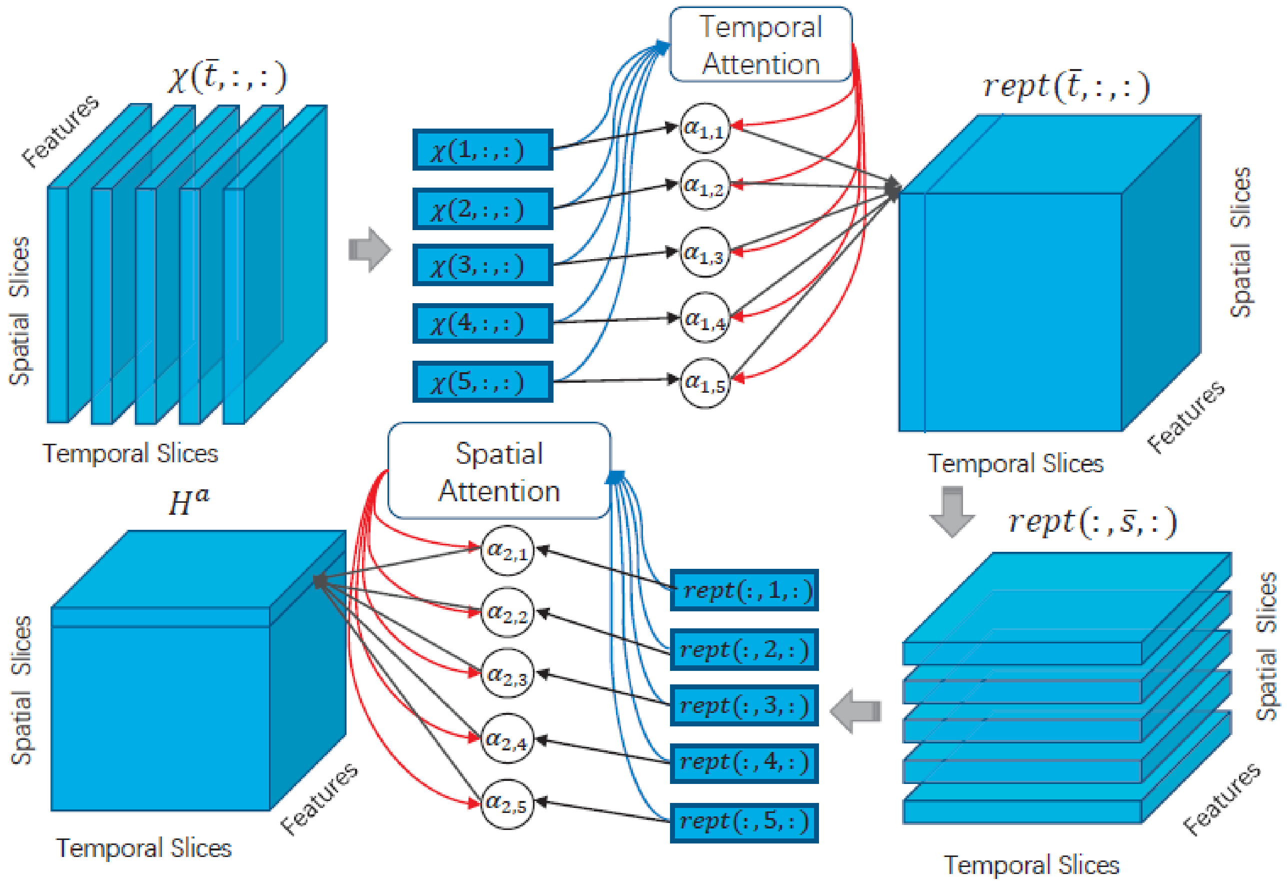
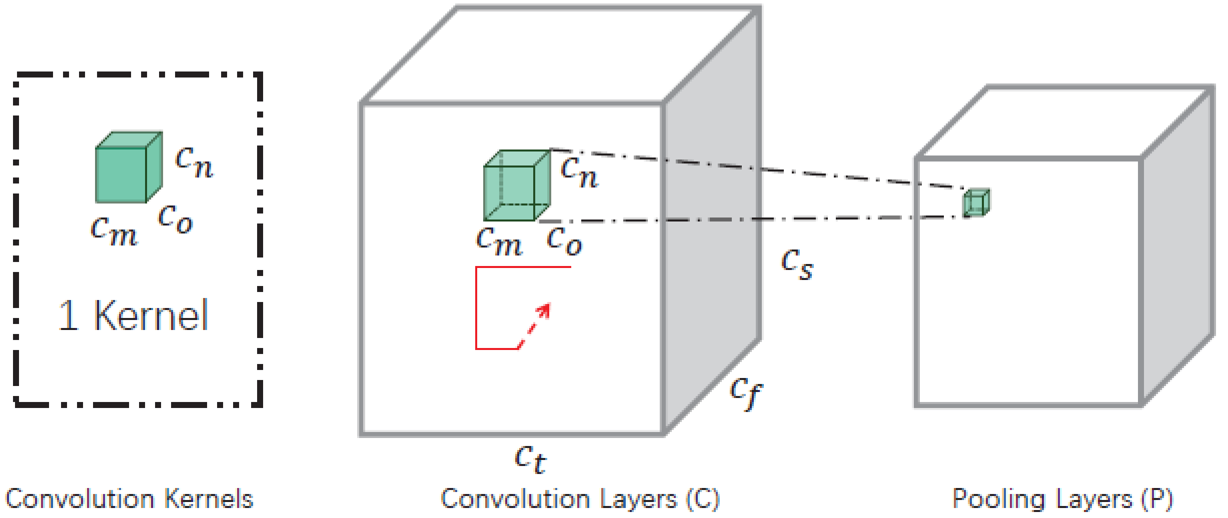


Fig. 6. Illustration of spatial-temporal attention neural networks.

3D Convolutional Layers



$$\mathcal{H}_j^c = \sum\nolimits_j \mathcal{H}^{c-1}(c_t - c_m, c_s - c_n, c_f - c_o)\mathcal{W}_i^c(c_m, c_n, c_o)$$

$$\mathcal{H}^c = \sigma\left(\sum\nolimits_j \mathcal{H}_j^c + b^c\right),$$

Detection Layer and Optimization

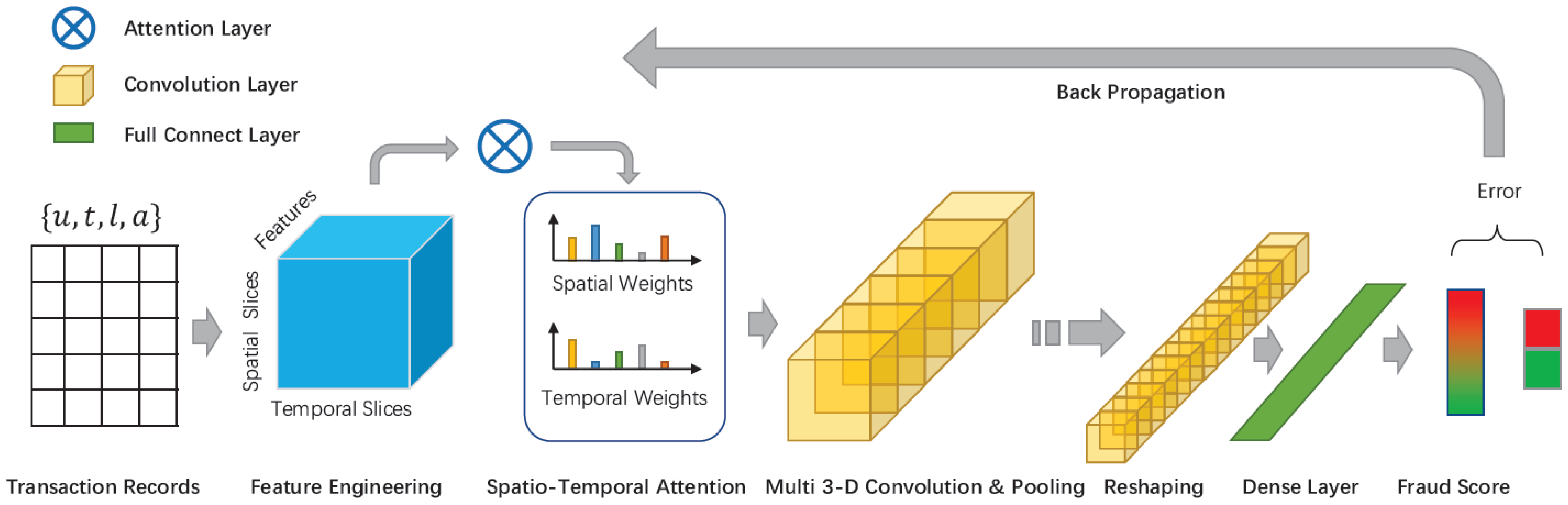
$$\mathcal{L}(\theta) = -\frac{1}{N}\sum\nolimits_{i=1}^{N} [y_i \log(\text{detect}(rep_i : \theta)) \\ + \lambda_3 (1 - y_i) \log(1 - \text{detect}(rep_i : \theta))]$$

where rep_i denotes the representation of the $i-th$ transaction record, which is the output of 3D ConvNet, and λ_3 indicates the sample weight according to the biased distribution of fraud and legitimate records; y_i denotes the label of $i-th$ records, which is set to 1 if the record is fraud and 0 otherwise; $\text{detect}(rep_i)$ is the detection function that maps rep_i to a real valued score, indicating the probability that whether the current transaction is fraudulent. We implement $\text{detect}(rep_i : \theta)$ with two-layer ReLU and one-layer sigmoid network.

TABLE 1
Result of the fraud detection experiment.

	AUC (Oct)	AUC (Nov)	AUC (Dec)
LR	0.7247	0.7163	0.7199
GBDT	0.7868	0.7949	0.7864
MLP	0.7803	0.8012	0.7891
Deep & Wide	0.8210	0.8197	0.8108
CNN-max	0.8352	0.8367	0.8267
AdaBM	0.8243	0.8249	0.8232
LSTM-seq	0.8368	0.8353	0.8290
STAGN-nograph	0.8435	0.8462	0.8509
STAGN-notrans	0.8413	0.8507	0.8596
STAN	0.8832	0.8789	0.8865
STAGN-notemp	0.8631	0.8604	0.8736
STAGN-nospat	0.8602	0.8531	0.8583
STAGN-no3d	0.8688	0.8629	0.8716
STAGN-all	**0.8973****	**0.8897****	**0.8983****

Spatio-Temporal Attention-Based Neural Network for Credit Card Fraud Detection



WWW2021

Multimodal and Contrastive Learning for Click Fraud Detection

Weibin Li*, Qiwei Zhong*, Qingyang Zhao, Hongchun Zhang, Xiaonan Meng
Alibaba Group
Hangzhou, China
{dece.lwb,yunwei.zqw,qingyang.zqy,hongchun.zhc,xiaonan.mengxn}@alibaba-inc.com

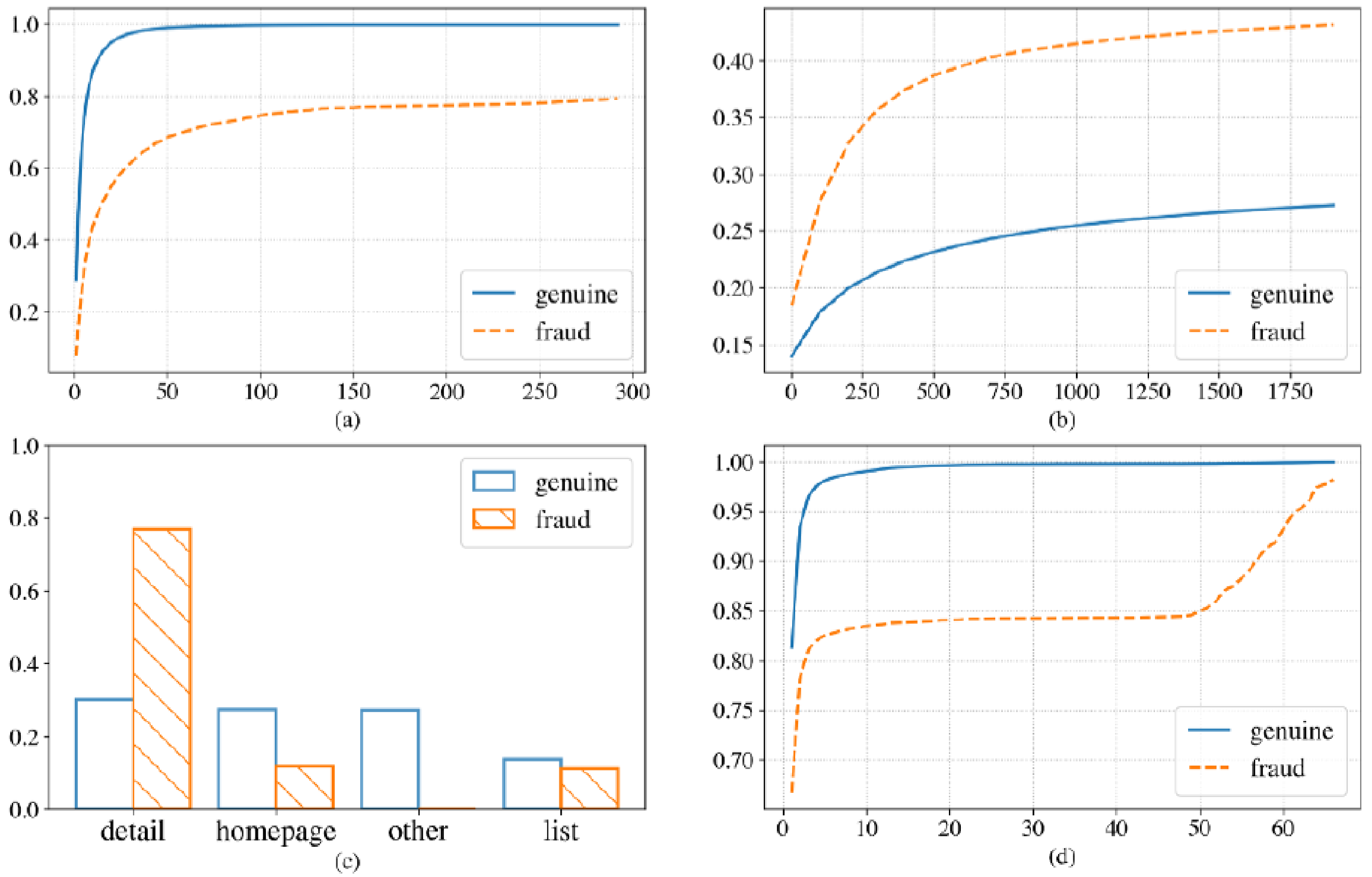
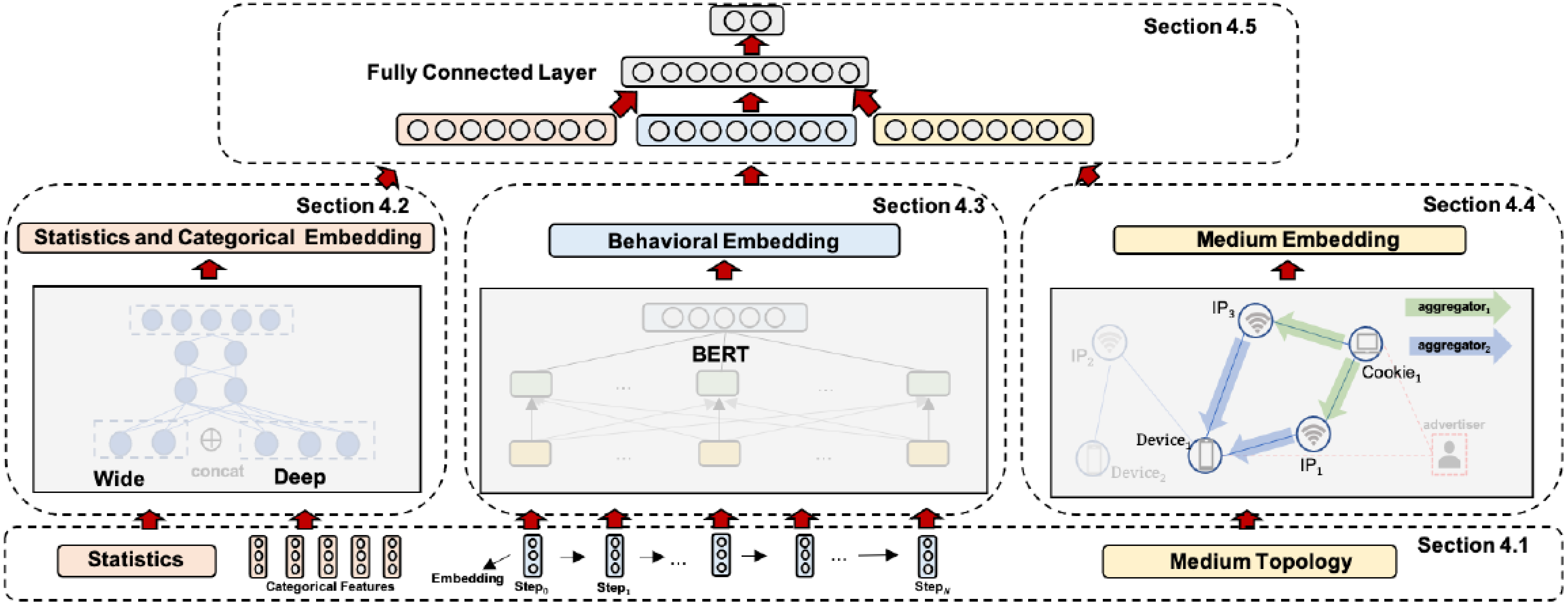


Figure 2: Statistical feature of fraudsters and genuine users: (a) cumulative distributions of the average number of clicks per IP; (b) cumulative distributions of the average time interval between the click time and the time that CookieID was generated; (c) page categorical properties of behaviors; (d) cumulative distributions of the average number of media.

- We propose a novel multimodal and contrastive learning network to solve this problem. Specifically, multimodal information including statistic and categorical features, behavior sequences and media relationships are involved to perform comprehensive click representations, and multiple layer perceptron is utilized to integrate them. Furthermore, to solve the imbalance problem, contrastive learning is elaborately exploited during training.

Field	**Description**
AbsPos	Absolute position of an ad on website
AdvertiserID	Unique identifier of advertiser
CdTime	Interval between display time and click time
ClickID	Unique identifier of a particular click
ClickTime	Timestamp of a given click
CookieID	Unique identifier of users
CookieTime	Timestamp that cookie was generated
DeviceID	Unique identifier of mobile users
IP	Public IP address of a click
KeywordID	Unique identifier of ad word
PageType	Homepage, Detail, ...


Section 4.5
Fully Connected Layer
Section 4.2
Statistics and Categorical Embedding
Wide
concat
Deep
Section 4.3
Behavioral Embedding
BERT
Section 4.4
Medium Embedding
IP3
IP2
aggregator1
aggregator2
Cookie1
Device1
Device2
IP1
advertiser
Section 4.1
Statistics
Categorical Features
Embedding
Step0
Step1
StepN
Medium Topology

Definition 1. Wide and Deep feature: wide features are continuous features in each click, including original values (e.g., CdTime), combined features (e.g., AbsPos and CdTime), and demographic features (e.g., the number of cookies in the last day of IP). Deep features are categorical features in each click, such as AdvertiserID, KeywordID. $x^{(w)} = [x_1^{(w)}, x_2^{(w)}, ..., x_l^{(w)}]$ is a vector of l-dimensional wide feature, and $x^{(d)} = [x_1^{(d)}, x_2^{(d)}, ..., x_r^{(d)}]$ is a vector of r-dimensional deep feature.

Definition 2. Behavior sequence: the sequence of pages visited by a user before the ad is clicked, such as "Homepage -> List -> Detail -> $\cdots$", as shown in Figure 3. $x^{(b)} = [x_1^{(b)}, x_2^{(b)}, ..., x_t^{(b)}]$ is a vector of t-dimensional behavior sequence. Specifically, the value of t in our model is 300.

4.1 Input layer

Every element in the sequences $x^{(d)}$ and $x^{(b)}$ for each click needs to be transferred into embedding. After looking up from two embedding matrices $W^{(d)}, W^{(b)}$ respectively, $x^{(d)}$ and $x^{(b)}$ are converted to $e^{(d)} = [e_1^{(d)}, e_2^{(d)}, ..., e_r^{(d)}]$, $e^{(b)} = [e_1^{(b)}, e_2^{(b)}, ..., e_t^{(b)}]$, of which each element is an embedding vector, as shown in Figure 3.

$$e^{(d)} = \text{LOOKUP}\left(W^{(d)}; x^{(d)}\right) \tag{1}$$

$$e^{(b)} = \text{LOOKUP}\left(W^{(b)}; x^{(b)}\right) \tag{2}$$

where LOOKUP $(W; x)$ is an operator to get vectors from W using each element of x as subscript. The embedding vectors are initialized randomly and then the values are trained with the model parameters to minimize the final loss function during training.

4.2 Wide and Deep Network

$$e^{(wd)} = \text{CONCAT}\left(e^{(d)}, x^{(w)}\right)$$

$$v^{(wd)} = \text{ReLU}\left(W_{wd}^{(L)} \ldots \text{ReLU}\left(W_{wd}^{(1)} e^{(wd)} + b_{wd}^{(1)}\right) + b_{wd}^{(L)}\right)$$

4.3 Behavior Sequence Network

$$v^{(b)} = \text{BERT}\left(e^{(b)}\right)$$

Definition 3. Multi-media heterogeneous network: given a graph $G = (V, E)$, the feature of the node $x^{(v)} = [x_1^{(v)}, x_2^{(v)}, ..., x_s^{(v)}]$ is a vector of s-dimensional feature integrated from itself and neighbors. The node types in our heterogeneous network are IP, CookieID, and DeviceID. For example, If a CookieID uses an IP to visit the website, the two nodes are neighbors, and an edge will be connected between them (as shown in Figure 3). For attributes of heterogeneous network, we collect 542 attributes for each medium (node), such as demographic information and click frequency. For each relation (link), we construct 90 attributes such as link type (e.g., click, login, and pay), first/last related time, and interaction frequency.

4.4 Multi-media Heterogeneous Network

$$h^k_{\mathcal{N}(v)} = \text{AGGREGATE}_k\left(\left\{h_u^{k-1}, \forall u \in \mathcal{N}(v)\right\}\right)$$

$$h_v^k = \sigma\left(\text{W}_v^k \cdot \text{CONCAT}\left(h_v^{k-1}, h^k_{\mathcal{N}(v)}\right)\right)$$

where h_v^k denotes a node's representation at this step. $\mathcal{N}(v)$ are all neighbor nodes of node v. Note that this aggregation step depends on the representations generated at the previous iteration, and representation $h_v^0 = x^{(v)}$ is defined as the input node features. We

Integration and Training

$$v^{(i)} = \left[v^{(wd)}; v^{(b)}; v^{(\breve{v})}\right]$$

$$z_2^{(i)} = W_2^{(i)}\left(\mathrm{ReLU}\left(W_1^{(i)} v^{(i)} + b_1^{(i)}\right)\right) + b_2^{(i)}$$

$$\hat{y} = \mathrm{softmax}\left(z_2^{(i)}\right)$$

contrastive learning

In this paper, we train our model with SOTA NT-Xent loss with regularization. Specifically, let multiple layer perceptron (MLP) be an encoder network mapping $z_2^{(i)}$ to the latent space z firstly.

$$z = \mathrm{MLP}\left(z_2^{(i)}\right) \tag{10}$$

Let $\mathrm{sim}(\boldsymbol{a}, \boldsymbol{b})$ denote the dot product between ℓ_2 normalized $\boldsymbol{a}$ and $\boldsymbol{b}$ (i.e. cosine similarity) in equation (11). When applied on a pair of positive samples $z^{(i)}$ and $z^{(j)}$ and other $2(N-1)$ negative examples, the loss function $\ell(i, j)$ for a positive pair of examples (i, j) is defined in equation (12).

$$\mathrm{sim}(\boldsymbol{a}, \boldsymbol{b}) = \boldsymbol{a}^\top \boldsymbol{b} / \|\boldsymbol{a}\| \|\boldsymbol{b}\| \tag{11}$$

$$\ell(i, j) = -\log \frac{\exp\left(\mathrm{sim}\left(z^{(i)}, z^{(j)}\right)/\tau\right)}{\sum_{k=1}^{2N} \not{1}_{[k \neq i]} \exp\left(\mathrm{sim}\left(z^{(i)}, z^{(k)}\right)/\tau\right)} \tag{12}$$

where $\not{1}_{[k \neq i]} \in \{0, 1\}$ is an indicator function evaluating to 1 if $k \neq i$, and τ denotes a temperature parameter. The final loss is computed across all positive pairs in a mini-batch.

$$\mathcal{L} = \frac{1}{2M} \sum_{k=1}^{M} [\ell(2k-1, 2k) + \ell(2k, 2k-1)] + \frac{\lambda}{2} \|\theta\|_2^2 \tag{13}$$

where λ is the regularization parameter and θ is the set of parameters of the proposed model.

Dataset	#Positive	#Negative	#Total	#Positive Rate
Training	276,956	2,265,022	2,541,978	10.89%
Testing	75,999	670,721	746,720	10.17%

Method	Precision	Recall	F1-score	AUC
Random Forest	0.867	0.403	0.550	0.685
LightGBM	0.892	0.416	0.567	0.686
GraphSAGE	0.973	0.545	0.699	0.785
BiLSTM	0.966	0.480	0.641	0.755
TextCNN	0.981	0.604	0.747	0.804
BERT	0.984	0.619	0.760	0.861
MCCF	**0.987**	**0.854**	**0.916**	**0.933**

Model	Precision	Recall	F1-score	AUC
MCCF$_{\backslash B}$	0.970	0.735	0.836	0.856
MCCF$_{\backslash V}$	0.975	0.776	0.864	0.882
MCCF$_{\backslash WD}$	0.979	0.807	0.884	0.905
MCCF$_{CE}$	0.985	0.832	0.902	0.918
MCCF	**0.987**	**0.854**	**0.916**	**0.933**