



Toward Anomaly Detection





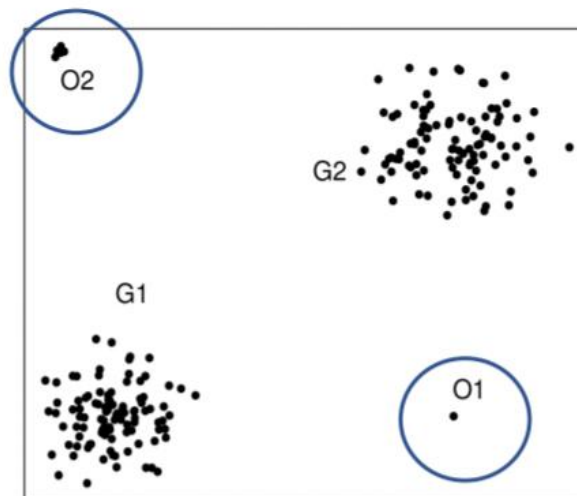
- 什么是异常?
- 异常检测的场景有哪些?
- 异常检测的难度在哪?
- 怎么做异常?





Anomalies are referred to as data objects that deviate significantly from the majority of data objects. —from DevNet, KDD 2019

- **Anomalies (a.k.a., outliers, novelties): Points that are significantly different from most of the data**
 - ✓ Rare
 - ✓ Irregular



Source: Wikipedia





Real-World Application Domains

Cybersecurity:

attacks, malware, malicious apps/URLs, biometric spoofing



Finance:

credit card/insurance frauds, market manipulation, money laundering, etc.



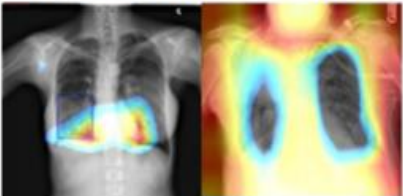
Social Network and Web Security:

false/malicious accounts, false/hate/toxic information



Healthcare:

lesions, tumours, events in IoT/ICU monitoring, etc.



Video Surveillance:

criminal activities, road accidents, violence, etc.



Industrial Inspection:

Defects, micro-cracks

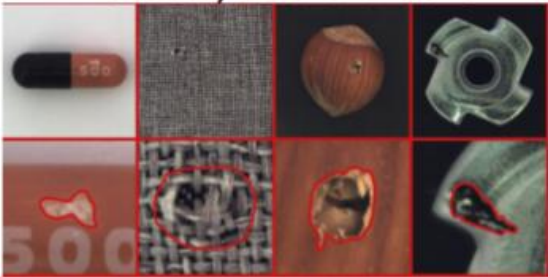


Image source: UCF-Crime data, MVTec AD data, etc.




异常检测的场景



(a) Normal Data (X-ray Scans)

(b) Normal + Abnormal Data (X-ray Scans)

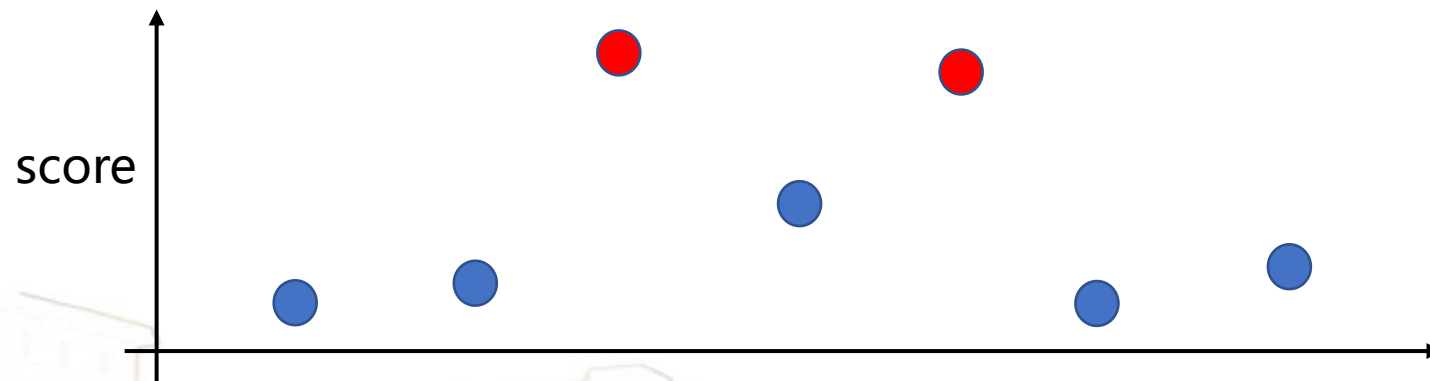
May be anomaly 

交易时间	委托数量	匹配数量	成交价格
2021-01-01 10:00:01	10000	8000	5.42
2021-01-01 10:00:01	4500	4500	4.98
2021-01-01 10:00:02	4000	2000	5.2
2021-01-01 10:00:03	2000000	5000	4.6
2021-01-01 10:00:03	300	300	5.16



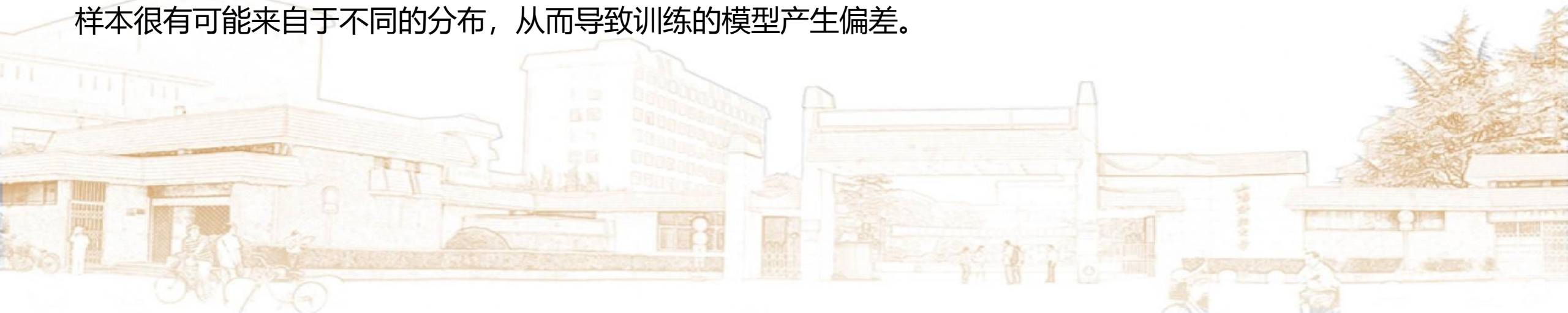
Model: $f(X)$ ——score (can be probability or any value greater than 0)

Evaluation: Higher score, more anomaly

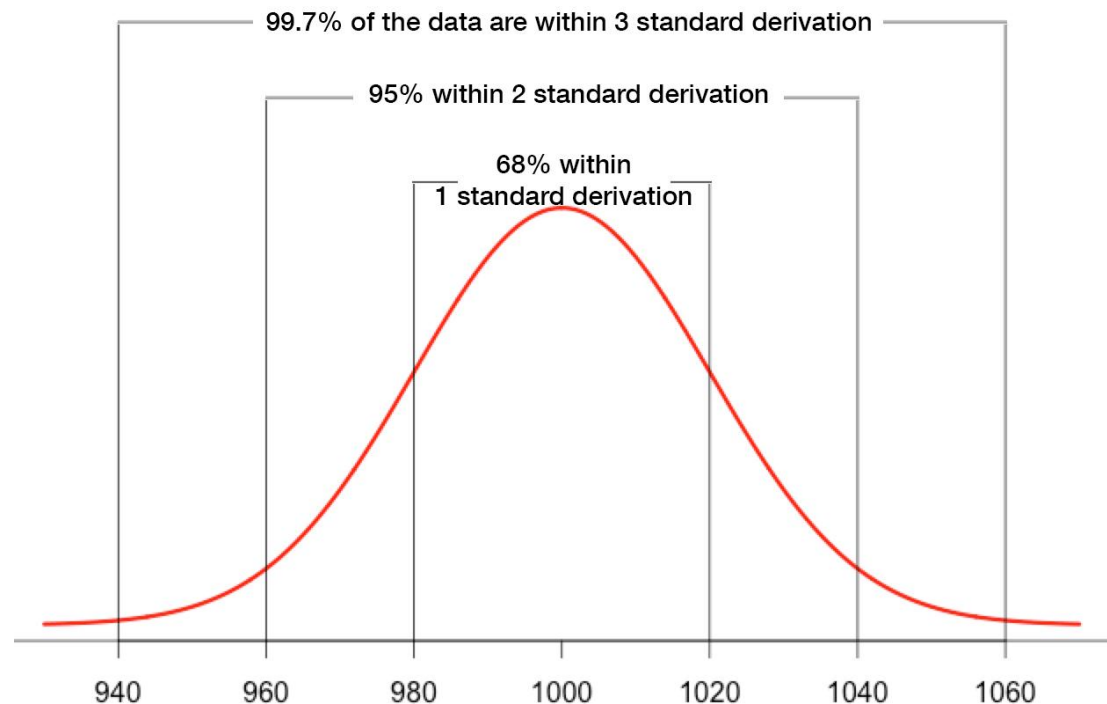




- **异常样本的稀疏性**——区分于大多数二分类以及多分类问题（样本分布较为平均），异常样本往往只占了总体样本的极少量部分，甚至 $<0.1\%$ ，即10000个样本中只有不到10个异常样本，我们需要训练合适的模型将极少量的异常样本检测出来。
- **异常样本的混淆性**——某些异常样本是具有混淆性的（故意与正常样本混淆），例如在股票交易场景中，内幕交易者往往会将自己伪装成正常交易的散户，利用分批、逐次递交委托单的形式完成最后的大额交易目的。
- **标记样本的匮乏性**——假设有10000个样本，人工将9990个正常样本与10个异常样本全部进行标注的成本是非常大的。
- **异常样本的多样性**——异常样本的特性往往是多样的，训练阶段的异常样本与测试阶段的异常样本很有可能来自于不同的分布，从而导致训练的模型产生偏差。



异常检测 (shallow model) —— Normal Distribution



一个最为直观、朴素的思想：样本X的某个维度下的特征是服从正态分布的，**处在尾部的样本大概率是异常样本。**

这样做会存在几个问题

- 无法有效处理高维特征
- 无法有效处理不服从正态分布的样本，即使采用混合高斯分布也无法有效建模
- 无法捕捉数据中其余的重要特征

异常检测 (shallow model) —— Isolation Forest



Isolation Forest (孤立森林) , by zhihua zhou et al., 2008:

- 孤立森林算法是基于 Ensemble 的异常检测方法，因此具有**线性的时间复杂度**。在处理大数据时速度快，所以目前在工业界的应用范围比较广。
- 孤立森林能够在Python sklearn库中被快速部署



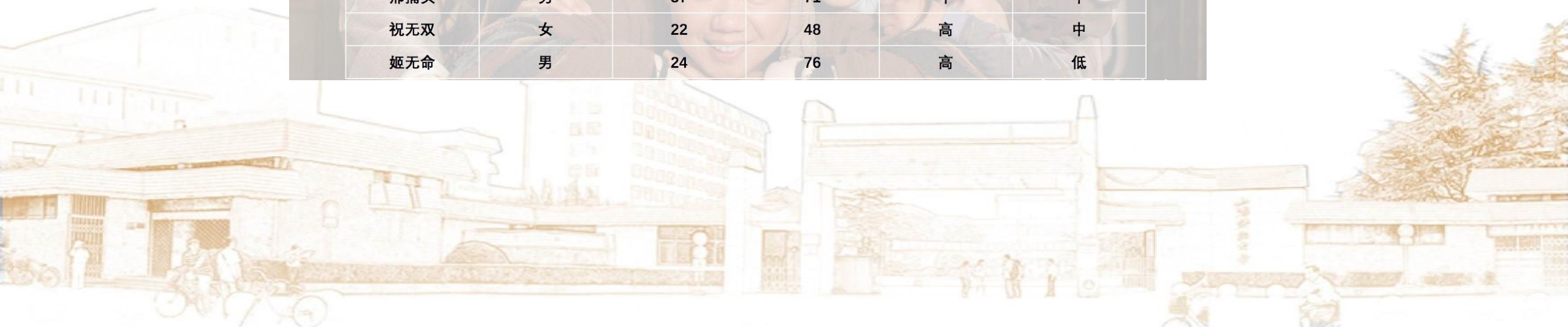
异常检测 (shallow model) ——Isolation Forest



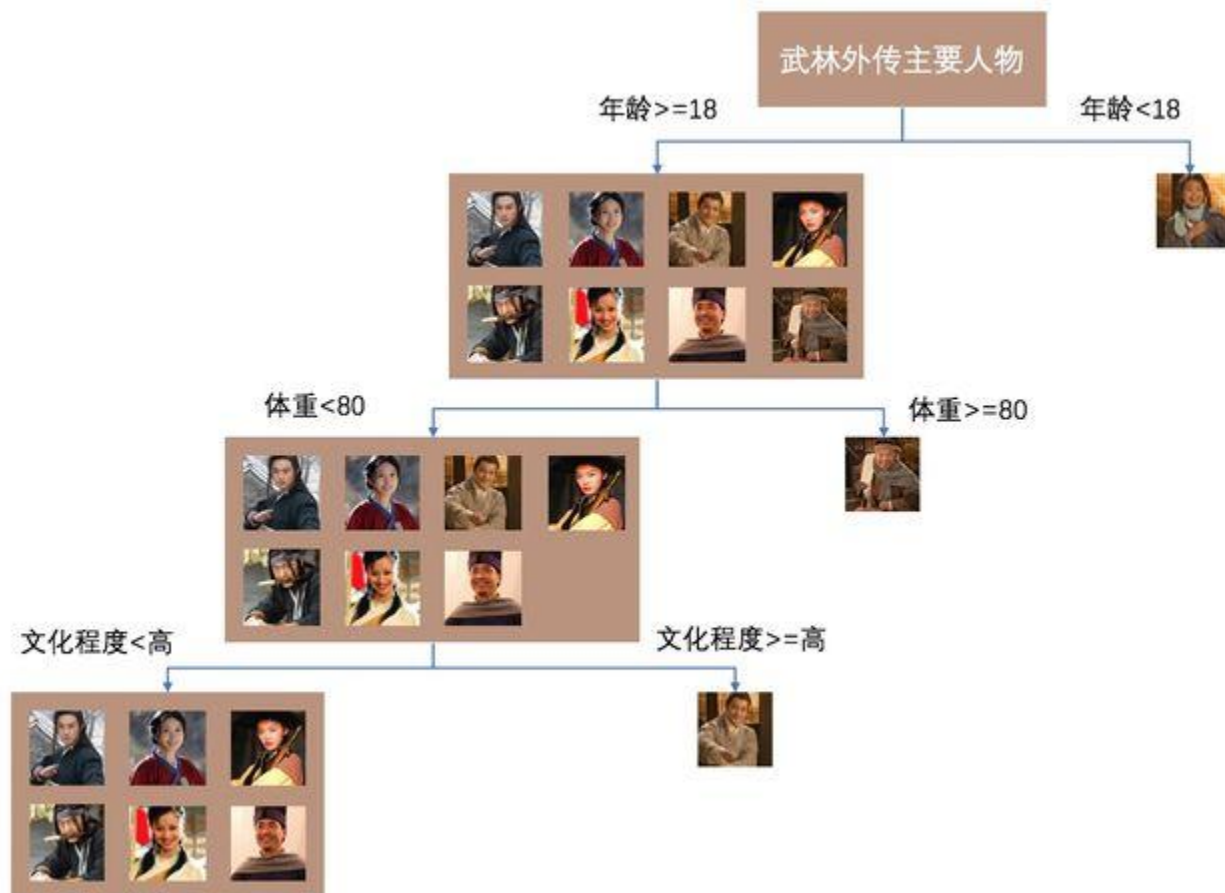
假设我们有一些样本X，我们需要将其中的异常点给识别出来：

武林外传主角基本数据

人物	性别	年龄	体重	武功水平	文化程度
佟湘玉	女	30	50	中	中
白展堂	男	25	70	高	中
郭芙蓉	女	22	49	高	中
吕轻侯	男	25	65	低	高
李大嘴	男	27	90	中	低
莫小贝	女	10	38	中	中
邢捕头	男	37	71	中	中
祝无双	女	22	48	高	中
姬无命	男	24	76	高	低



异常检测 (shallow model) —— Isolation Forest



- 异常程度的定义：**所分样本距离根节点的距离，越近越异常**；即越异常的样本点越容易通过一些特征（年龄、体重等），与正常样本区分出来。
- 划分的特征是随机的，即年龄 ≥ 20 、或者武力值为切分点也是有概率产生的。
- 最后的结果通过树的集成（此处只展示了一棵树），以增强模型的稳定性与收敛性。



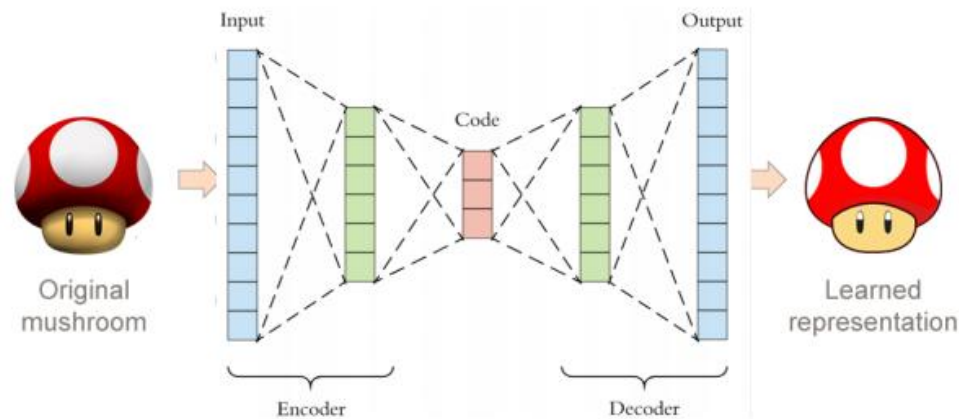
Shallow model的劣势 (why we use deep learning model?)



- **Weak capability of capturing intricate relationships**——shallow model往往只能捕捉数据内部一些浅层的特征，对于数据内部一些复杂特性的建模可能存在问题。而异常样本往往在高度非线性、超高维空间中才能被部分区分。
- **Lots of hand-crafting of algorithms and features** ——shallow model的效果往往与复杂的特征工程以及算法本身构造有着极大的关联，这会导致需要投入大量的人工成本完成模型的构建。



异常检测 (deep learning) : 从一个简单的auto-encoder (自编码器) 开始



The main idea of auto-encoder: 训练模型 (神经网络) 重构输入样本X, 使得重构样本与输入样本尽可能相近



异常检测 (deep learning) : 从一个简单的自编码器开始



Q:

- How to use the auto-encoder for anomaly detection?



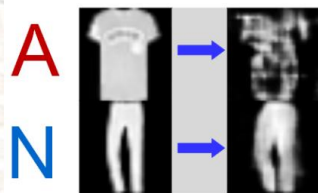
异常检测 (deep learning) : 从一个简单的自编码器开始



- How to use the auto-encoder for anomaly detection?

1. **在训练阶段, 只将正常样本输入模型**——训练阶段完成后, 得到的模型只知道如何正确的重构正常样本 (裤子)

2. 在测试阶段, 正常样本 (裤子) **能够被较好的重构 = 重构误差小 = 异常得分低**; 异常样本 (衣服) 由于模型没见过、没学过, 因此**不能够被较好的重构 = 重构误差大 = 异常得分高**。因此衣服能够成功的被检测为异常样本



异常检测 (deep learning) : 从一个简单的自编码器开始



Q:

- What is the major difference between isolation forest and auto-encoder? (从样本的角度回答)



异常检测：能否进一步提升auto-encoder?



Auto-encoder的思路是清晰的：**只利用正常样本训练模型，使模型只捕捉正常样本的特性，从而对于异常样本重构失败，以此进行检测**

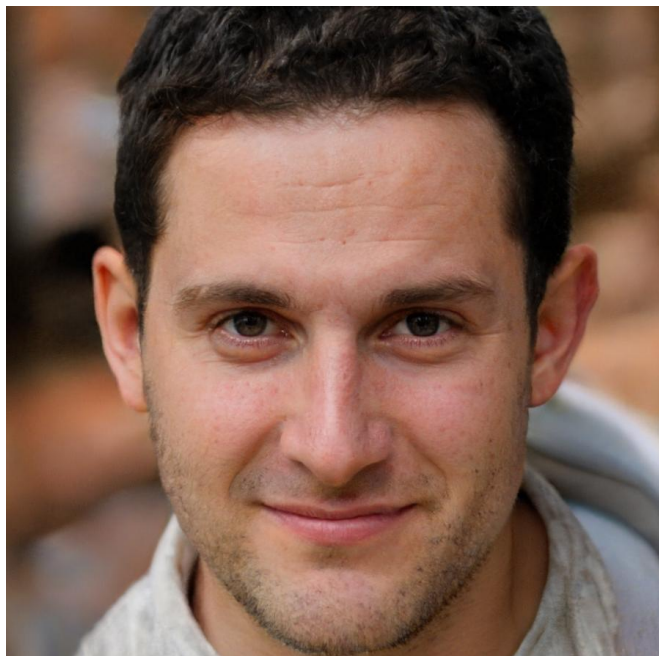
能否进一步提升？： we use GAN (Generative Adversarial Networks)!



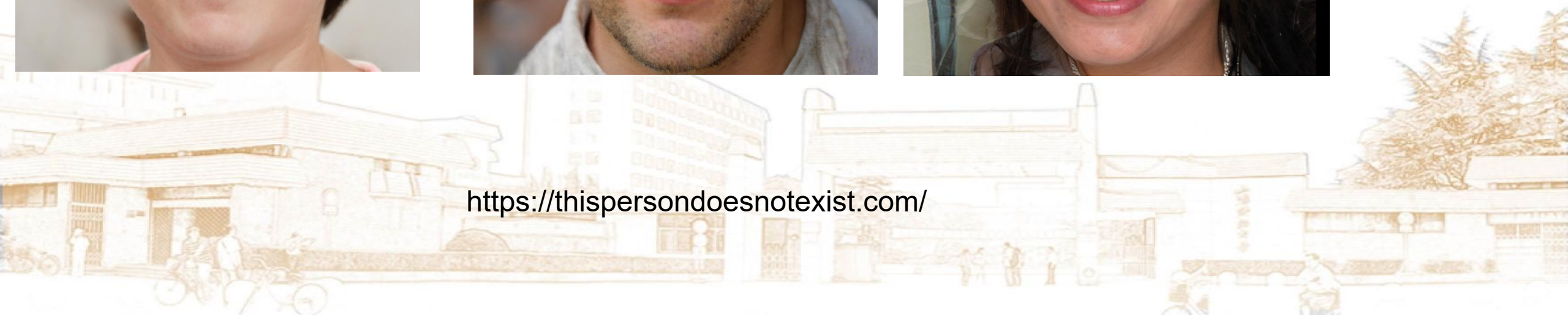
异常检测：能否进一步提升auto-encoder? ——we use GAN!



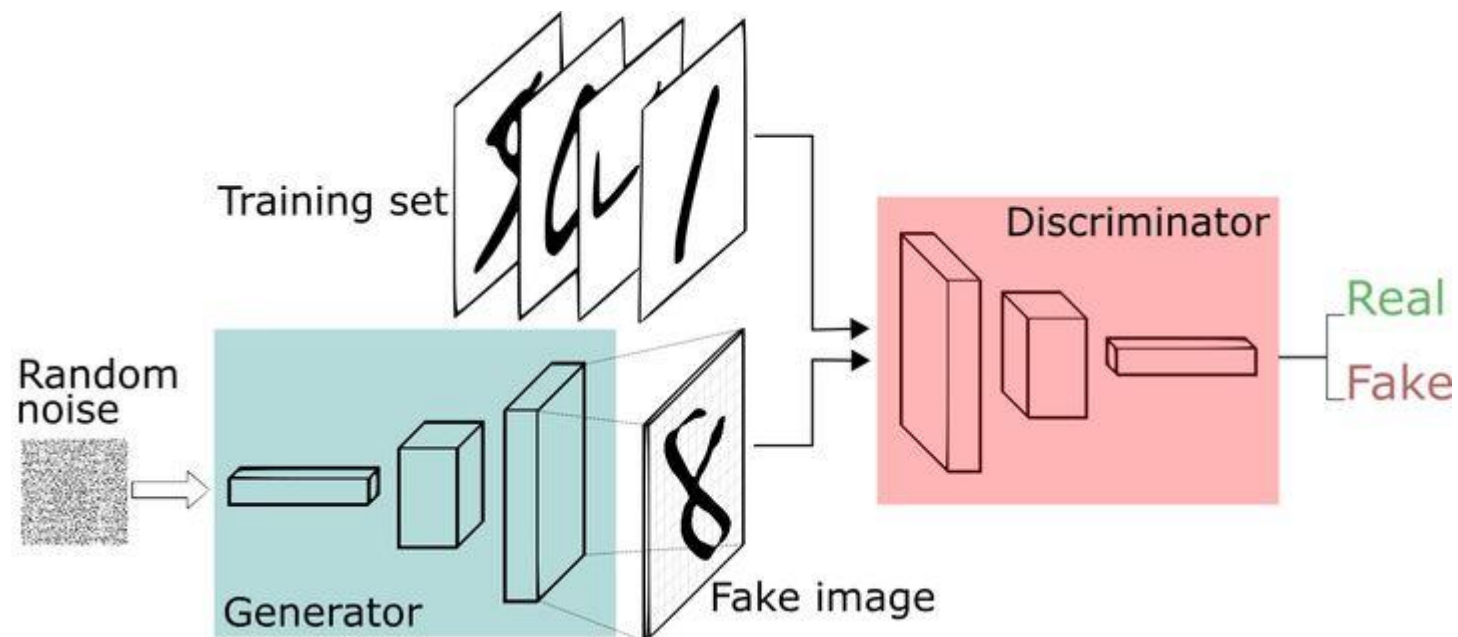
GAN (Generative Adversarial Networks), first proposed by Goodfellow, 2014



<https://thispersondoesnotexist.com/>



异常检测：能否进一步提升auto-encoder? ——we use GAN!



GAN is trained by a min-max game, and get a stable training stage until Nash equilibrium:

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\text{data}}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_z(z)} [\log(1 - D(G(z)))].$$

Try a GAN demo by the website——<https://poloclub.github.io/ganlab/>

异常检测：进阶版的auto-encoder——GANomaly (by Samet Akcay, 2018 ACCV)

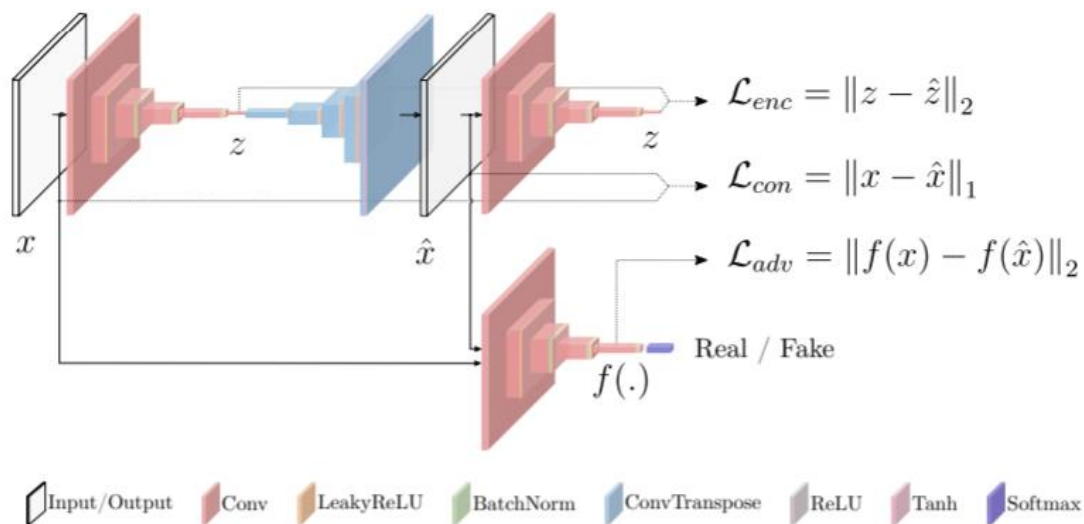
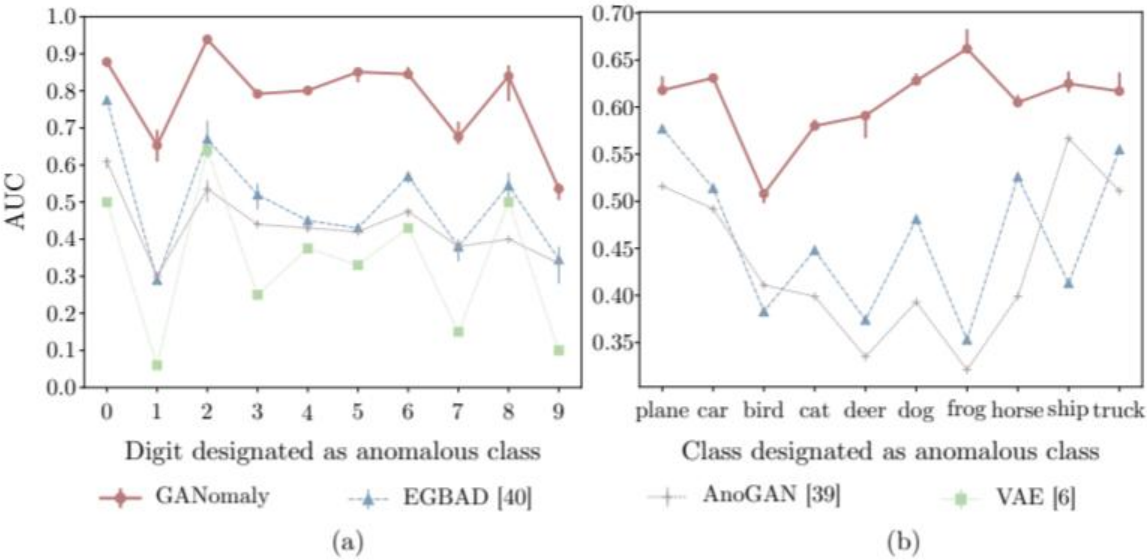


Fig. 2. Pipeline of the proposed approach for anomaly detection.

GANomaly vs Auto-encoder

- Auto-encoder only has encoder-decoder; GANomaly has **encoder-decoder-encoder** structure
- GANomaly additionally implements a **discriminator** for improving model performance

异常检测：进阶版的auto-encoder——GANomaly

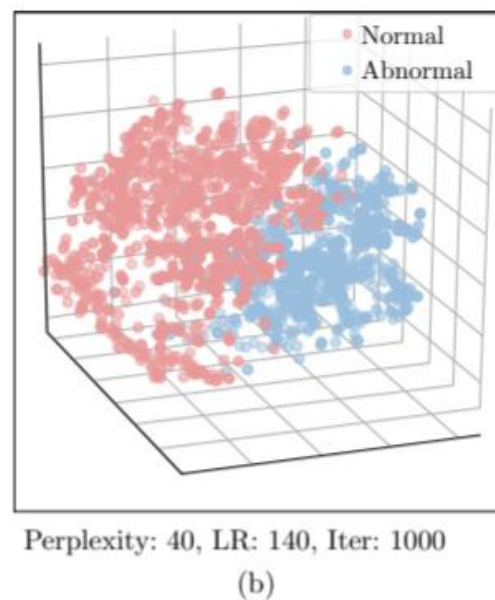
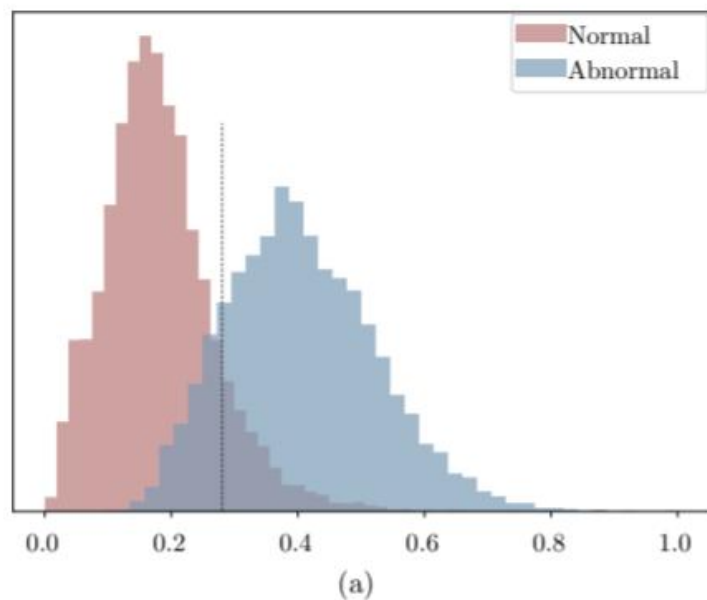


- 实验证明了 GANomaly 相较于 VAE (Variational Auto-Encoder) 模型以及其余 GAN 模型 (例如 AnoGAN 以及 EGBAD) 而言在多个数据集中均具有优势。

Method	UBA				FFOB
	gun	gun-parts	knife	overall	full-weapon
AnoGAN [39]	0.598	0.511	0.599	0.569	0.703
EGBAD [40]	0.614	0.591	0.587	0.597	0.712
GANomaly	0.747	0.662	0.520	0.643	0.882

Table 1. AUC results for UBA and FFOB datasets

异常检测：进阶版的auto-encoder——GANomaly



- GANomaly产生的得分能够很好的在正常样本与异常样本之间产生分布的差异性
- GANomaly对于正常样本与异常样本的embedding也能有效区分



金融风控案例：用GANomaly做异常股票交易检测

TRADEDATE	CUSTOMER	FUNCTION	FUNCTION_NAME		
日期	客户编号	指标编号	指标名称		
20170518	31904810	1.2E+09	大额频繁申报后大量撤单监控		
20170522	44487750	1.2E+09	集合竞价期间大额高（低）价申报监控		
20170510	6180339	1.2E+09	大额频繁申报后大量撤单监控		
20170524	31904810	1.2E+09	大额频繁申报后大量撤单监控		
20170504	82141659	1.2E+09	价格异常申报监控		
20170523	97508770	1.2E+09	大额频繁申报后大量撤单监控		
20170515	38631270	1.2E+09	营业部大量集中买卖同一股票		
20170515	44496540	1.2E+09	集合竞价期间大额高（低）价申报监控		
20170526	75763900	1.2E+09	涨跌幅限制大额申报		
20170517	6264539	1.2E+09	涨跌幅限制大额申报		
20170505	51973989	1.2E+09	大额频繁申报后大量撤单监控		
20170523	53893470	1.2E+09	单证券一段期间大量连续交易		
20170516	21958899	1.2E+09	集合竞价期间大额高（低）价申报监控		
20170515	73760760	1.2E+09	开放式基金频繁短期交易		
20170502	24665340	1.2E+09	大额频繁申报后大量撤单监控		
20170524	27687680	1.2E+09	单证券一段期间大量连续交易		
20170524	55712159	1.2E+09	集合竞价期间大额高（低）价申报监控		
20170510	94904479	1.2E+09	大额频繁申报后大量撤单监控		
20170505	93386289	1.2E+09	单证券一段期间大量连续交易		
20170504	63612719	1.2E+09	大量频繁高买低卖交易		

OCUR_DATE	SUB_MARKET	BRAN_CODE	MARKET_CODE	CENTRUST	CAPITAL_ACCOUNT	CURRENT_CURRENCY	STOCK_CODE	STOCK_TYPE	ORDER_QTY	MATCH_QTY	MATCH_SLIP	CUSTOMER	ORDER_DATE	ORDER_TIME	ORDER_PRICE	ORDER_TYPE	
日期	营业部	分支	市场代码	委托方式	资金账户	总部资金	币种	证券代码	证券类型	委托数量	匹配数量	成交金额	客户编号	委托日期	委托时间	委托价格	委托类型
20170502	2201	1	3	99994	0	0	603889	0	1500	0	0	17839929	20170502	14:43:23	15.23	0	
20170502	2201	1	3	99994	0	0	603023	0	3000	3000	36450	17839929	20170502	14:44:41	12.15	0	
20170502	8039	1	j	93037	0	0	732180	0	17000	0	0	72351989	20170502	10:35:18	27.85	0	
20170502	8039	1	j	4993	0	0	732180	0	17000	0	0	72351989	20170502	10:34:59	27.85	0	
20170502	8039	2	j	4993	0	0	2869	0	11500	0	0	72351989	20170502	10:35:03	21.8	0	
20170502	8039	2	j	4993	0	0	2871	0	17000	0	0	72351989	20170502	10:35:08	15.39	0	
20170502	8065	2	j	1353	0	0	300035	0	1000	1000	14180	4619809	20170502	9:33:26	14.18	0	
20170502	8065	1	j	1353	0	0	732180	0	5000	0	0	4619809	20170502	9:34:27	27.85	0	
20170502	8065	2	j	1353	0	0	2126	0	1000	1000	10140	4619809	20170502	9:38:53	10.14	0	
20170502	8065	1	j	1353	0	0	732180	0	5000	0	0	4619809	20170502	9:41:13	27.85	0	
20170502	8065	2	8	1353	0	0	2126	0	1000	1000	10480	4619809	20170502	14:49:43	10.48	0	
20170502	8065	2	8	1353	0	0	300035	0	1000	1000	14070	4619809	20170502	14:50:47	14.07	0	
20170502	2201	1	8	10393	0	0	732180	0	6000	0	0	17753859	20170502	14:28:05	27.85	0	
20170502	2201	2	8	10393	0	0	2869	0	11500	0	0	17753859	20170502	14:28:05	21.8	0	
20170502	2201	2	8	10393	0	0	2871	0	17000	0	0	17753859	20170502	14:28:05	15.39	0	
20170502	2201	1	j	99994	0	0	603023	0	2500	0	0	17839929	20170502	9:21:42	12.41	0	
20170502	2201	1	j	99994	0	0	603023	0	3500	0	0	17839929	20170502	9:21:54	12.55	0	
20170502	2201	1	j	99994	0	0	603023	0	2500	0	0	17839929	20170502	9:22:00	12.53	0	
20170502	2201	1	j	99994	0	0	603023	0	3500	0	0	17839929	20170502	9:22:11	12.67	0	
20170502	2201	1	j	99994	0	0	603023	0	3500	0	0	17839929	20170502	9:22:18	12.7	0	

异常的、违规的股票交易是层出不穷的...一个好的异常算法既需要检测出已知的异常，也需要对未来可能出现的异常具备良好的检测效果。

对于time-series data，把FC layer换为RNN layer能够有效捕捉sequential的信息！

异常检测：能否进一步提升？——利用有限的异常样本



- GANomaly在auto-encoder的基础上进行了提升：利用encoder-decoder-encoder的结构+discriminator的对抗训练增强了模型的性能。
- GANomaly的建模过程仍然是基于大量的正常样本，但是**现实情况中收集大量标注的正常样本可能是困难的，能否利用极少量的异常样本快速提升模型性能？**
- 分享一个近期我们做的工作：Weakly-supervised Generative Adversarial Network (WSGAN)，区别于GANomaly，WSGAN只需要**少量的异常样本+大量无标签样本**即可快速提升模型检测性能



异常检测：能否进一步提升？——利用有限的异常样本

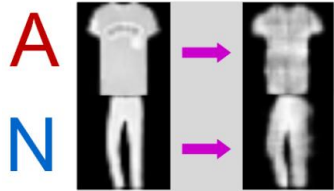


模型 \ 对比指标	IF	OCSVM	GANomaly	WSGAN	RF	XGBoost
F1 (5%)	9.8	6.2	4.2	55.2	5.8	5.8
F1 (10%)	12.9	4.5	25.6	46.2	6.7	6.0
F1 (15%)	11.0	3.4	33.4	34.4	5.5	4.9

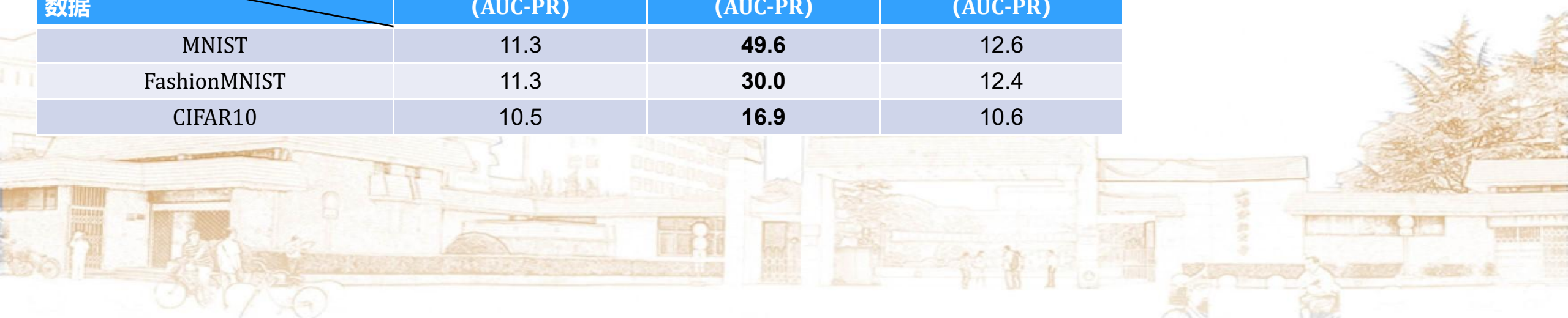
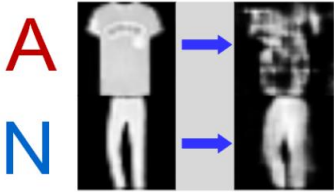
模型 \ 对比指标	IF	OCSVM	GANomaly	WSGAN	RF	XGBoost
AUC-ROC	85.6	77.3	75.5	90.0	64.5	82.7
AUC-PR	35.4	20.3	25.6	63.8	20.2	39.9

数据 \ 模型	GANomaly (AUC-PR)	WSGAN (AUC-PR)	ResNet18 (AUC-PR)
MNIST	11.3	49.6	12.6
FashionMNIST	11.3	30.0	12.4
CIFAR10	10.5	16.9	10.6

样本重构过程
(GANomaly)



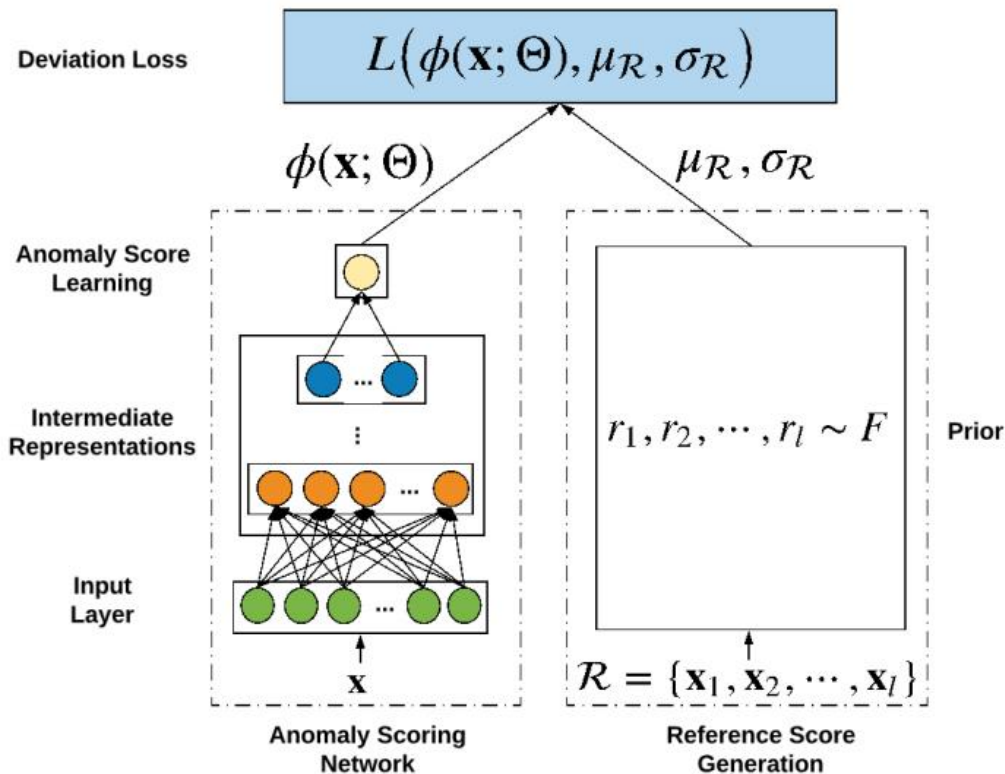
样本重构过程
(WSGAN)



异常检测：能否进一步提升？——end-to-end structure



DevNet, Guansong Pang et al. KDD 2019



- 另外一种对于异常检测进行建模的方式在于构建端到端（end-to-end）的模型，这种方法训练神经网络直接学习映射 $f(X)=score$ ，而不是采用auto-encoder或者GANomaly的two-stage方法，先重构，再计算score。
- End-to-end方式理论上能获得更好的效果，模型的训练效果与检测性能关联性更强。

异常检测：能否进一步提升？——end-to-end structure



Table 1: AUC-ROC and AUC-PR Performance (with \pm standard deviation) of DevNet and Four Competing Methods. #obj. is the overall data size, D is the dimensionality size, f_1 and f_2 denote the percentage that the labeled anomalies respectively comprise in the training data and the total anomalies. D in *URL* and *news20*, i.e., ‘3M’ and ‘1M’, are short for 3,231,961 and 1,355,191, respectively. The best performance is boldfaced.

Data Characteristic					AUC-ROC Performance					AUC-PR Performance				
Data	#obj.	D	f_1	f_2	DevNet	REPEN	DSVDD	FSNet	iForest	DevNet	REPEN	DSVDD	FSNet	iForest
donors	619,326	10	0.01%	0.08%	1.000 \pm 0.000	0.975 \pm 0.005	0.995 \pm 0.005	0.997 \pm 0.002	0.874 \pm 0.015	1.000 \pm 0.000	0.508 \pm 0.048	0.846 \pm 0.114	0.994 \pm 0.002	0.221 \pm 0.025
census	299,285	500	0.01%	0.16%	0.828 \pm 0.008	0.794 \pm 0.005	0.835 \pm 0.014	0.732 \pm 0.020	0.624 \pm 0.020	0.321 \pm 0.004	0.164 \pm 0.003	0.291 \pm 0.008	0.193 \pm 0.019	0.076 \pm 0.004
fraud	284,807	29	0.01%	6.10%	0.980 \pm 0.001	0.972 \pm 0.003	0.977 \pm 0.001	0.734 \pm 0.046	0.953 \pm 0.002	0.690 \pm 0.002	0.674 \pm 0.004	0.688 \pm 0.004	0.043 \pm 0.021	0.254 \pm 0.043
celeba	202,599	39	0.02%	0.66%	0.951 \pm 0.001	0.894 \pm 0.005	0.944 \pm 0.003	0.808 \pm 0.027	0.698 \pm 0.020	0.279 \pm 0.009	0.161 \pm 0.006	0.261 \pm 0.008	0.085 \pm 0.012	0.065 \pm 0.006
backdoor	95,329	196	0.04%	1.29%	0.969 \pm 0.004	0.878 \pm 0.007	0.952 \pm 0.018	0.928 \pm 0.019	0.752 \pm 0.021	0.883 \pm 0.008	0.116 \pm 0.003	0.856 \pm 0.016	0.573 \pm 0.167	0.051 \pm 0.005
URL	89,063	3M	0.04%	1.69%	0.977 \pm 0.004	0.842 \pm 0.006	0.908 \pm 0.027	0.786 \pm 0.047	0.720 \pm 0.032	0.681 \pm 0.022	0.103 \pm 0.003	0.475 \pm 0.040	0.149 \pm 0.076	0.066 \pm 0.012
campaign	41,188	62	0.10%	0.65%	0.807 \pm 0.006	0.723 \pm 0.006	0.748 \pm 0.019	0.623 \pm 0.024	0.731 \pm 0.015	0.381 \pm 0.008	0.330 \pm 0.009	0.349 \pm 0.023	0.193 \pm 0.012	0.328 \pm 0.022
news20	10,523	1M	0.37%	5.70%	0.950 \pm 0.007	0.885 \pm 0.003	0.887 \pm 0.000	0.578 \pm 0.050	0.328 \pm 0.016	0.653 \pm 0.009	0.222 \pm 0.004	0.253 \pm 0.001	0.082 \pm 0.010	0.035 \pm 0.002
thyroid	7,200	21	0.55%	5.62%	0.783 \pm 0.003	0.580 \pm 0.016	0.749 \pm 0.011	0.564 \pm 0.017	0.688 \pm 0.020	0.274 \pm 0.011	0.093 \pm 0.005	0.241 \pm 0.009	0.116 \pm 0.014	0.166 \pm 0.017
Average					0.916 \pm 0.004	0.838 \pm 0.006	0.888 \pm 0.011	0.750 \pm 0.028	0.708 \pm 0.018	0.574 \pm 0.008	0.263 \pm 0.010	0.473 \pm 0.025	0.270 \pm 0.037	0.140 \pm 0.015
P-value					-	0.004	0.023	0.004	0.004	-	0.004	0.004	0.004	0.004

金融风控案例：用DevNet有效检测信用卡欺诈案例



频率	时间维度	变量维度
最大	交易前10天	交易笔数
平均	交易前5天	入账交易笔数
累计	交易前1天	出账交易笔数
	交易当天至交易时刻	同交易对手交易笔数
	交易前10小时	同交易对手入账笔数
	交易前5小时	同交易对手出账笔数
	交易前3小时	交易对手个数
	交易前1小时	入账交易对手个数
	交易前30分钟	出账交易对手个数
	交易前10分钟	交易金额
	交易前1分钟	入账交易金额
		出账交易金额
		同交易对手交易金额
		同交易对手入账金额
		同交易对手出账金额

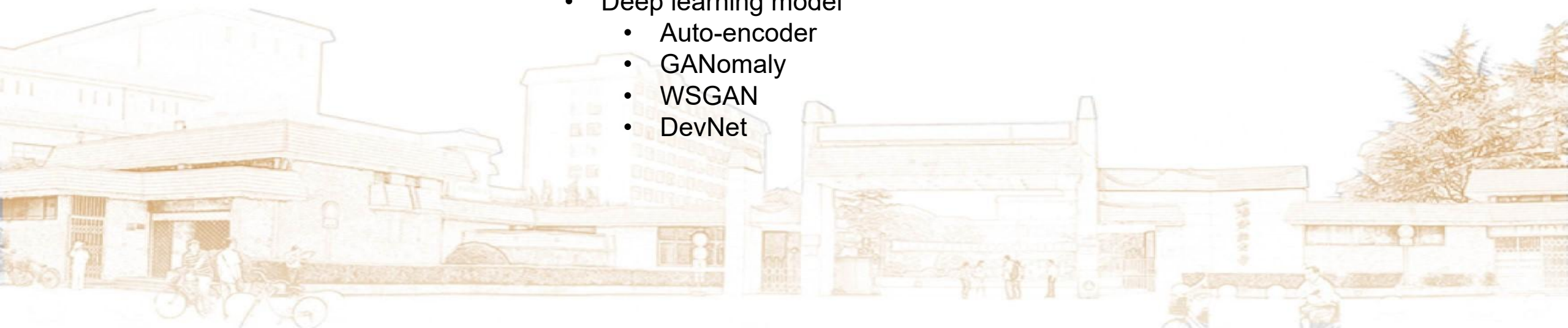
模型的输入数据可以用构建衍生变量的形式捕捉宝贵的时序信息，也可以直接利用时序数据，结合能够捕捉序列信息的模型（例如RNN以及Transformer），为下游异常检测任务提供强有力的backbone。

已知的欺诈案例往往是少量、宝贵的，数据类别极端不平衡。合适的损失函数（例如deviation loss）能够显著提升模型效果！





- **What is anomaly?**
- **The applications of anomaly detection?**
- **The difficulty of anomaly detection?**
- **How to detect anomalies?**
 - Shallow model
 - Normal distribution
 - Isolation Forest
 - Deep learning model
 - Auto-encoder
 - GANomaly
 - WSGAN
 - DevNet



参考资料

Python Package: PyOD

Anomaly Detection Resources: <https://github.com/yzhao062/anomaly-detection-resources>

Paper:

Deep SVDD: Ruff, Lukas, et al. "Deep one-class classification." *International conference on machine learning*. PMLR, 2018.

GANomaly: Akcay, Samet, Amir Atapour-Abarghouei, and Toby P. Breckon. "Ganomaly: Semi-supervised anomaly detection via adversarial training." *Asian conference on computer vision*. Springer, Cham, 2018.

Deep SAD: Ruff, Lukas, et al. "Deep semi-supervised anomaly detection." *arXiv preprint arXiv:1906.02694* (2019).

DevNet: Pang, Guansong, Chunhua Shen, and Anton van den Hengel. "Deep anomaly detection with deviation networks." *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 2019.





Q&A

Open question: 上述模型 (Auto-encoder, GANomaly, DevNet) 都存在一个普遍问题——**对于样本分布外的异常识别效果不好**。换句话说，猫和狗中狗是异常，模型学到了这点。新来一个老虎，模型并不能有效检测出老虎也是异常，尽管老虎与正常样本的猫还是有很大不同，如何解决？



SUFE AI Lab——系列AD工作



- 有这么多的异常检测算法，究竟哪个好？

ADbench: Anomaly detection benchmark, NeurIPS 2022

- 知道了现有算法的问题，如何改进？

Anomaly Detection with Score Distribution Discrimination, KDD 2023

- 能否做异常检测算法的自动化选择？

ADGym: Design Choices for Deep Anomaly Detection, NeurIPS 2023

