Beijing Forest Studio
北京理工大学信息系统及安全对抗实验中心

深度学习系统安全性测试及测试样本优先级排序

硕士研究生　王若辉

2021 年 11 月 28 日

内容提要

- 背景简介
- 历史发展
- 基本概念
- 算法原理
- 应用总结
- 参考文献

内容提要

- 预期收获
  - 1.了解深度学习系统安全性测试的发展历史。
  - 2.了解测试样本优先级排序的基本概念。
  - 3.了解测试样本优先级排序的最新研究。

提问

- 提问
  - 模糊测试为什么无法直接应用于深度学习系统测试?
  - 测试样本优先级排序任务是如何解决深度学习系统测试中两个关键问题的?
  - **Prima方法为何构建模型排序器?**
    - 而不是构建模型分类器，在通过预测概率进行排序?
  - **Prima方法为何使用RAUC而不是使用APFD?**

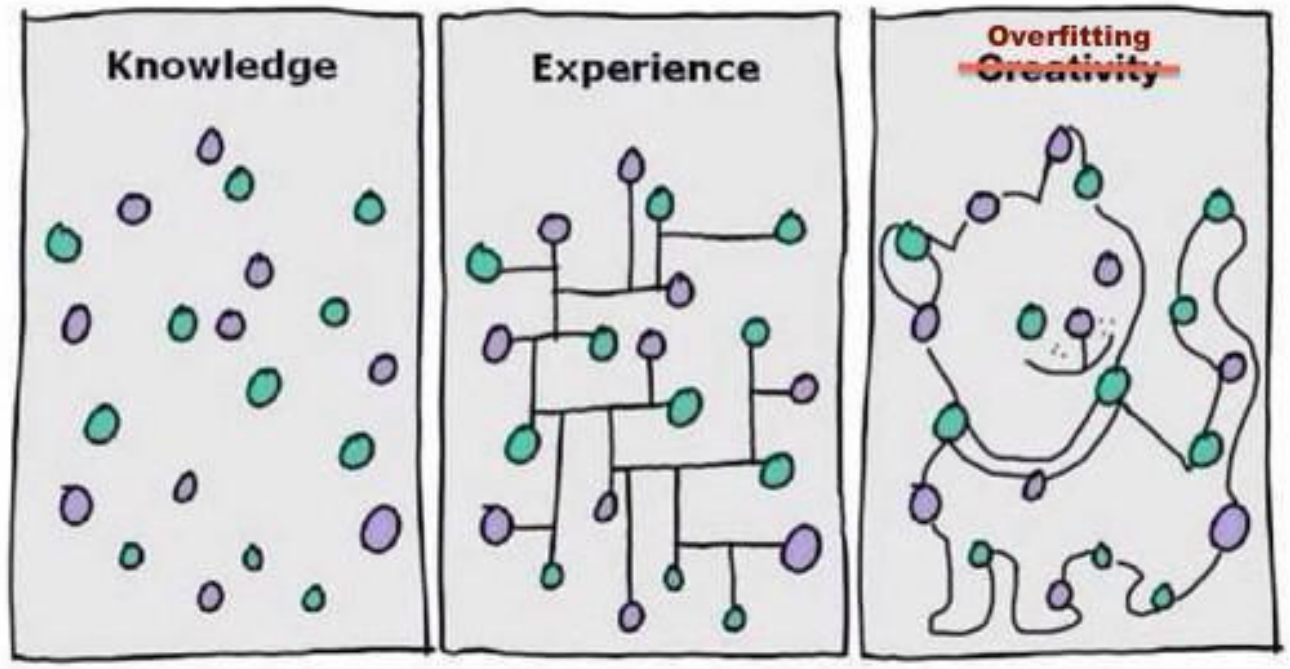
背景介绍

- 深度学习在近十年来取得了长足发展
- 在复杂领域
  - 图像识别
  - 语音识别
  - 入侵检测

    ……
- 表现出等同或远超于人类的性能而逐渐被集成到软件体系中形成深度学习系统并被广泛应用

背景介绍

- 这一方面，推动了深度学习的发展；另一方面，也对深度学习的安全性提出了巨大挑战
  - 由于模型过拟合、欠拟合，训练数据不平衡等问题，模型在面对极端样本时往往会做出不正确的预测行为



背景介绍

- 在对预测正确性至关重要的领域
  - 如自动驾驶、恶意流量检测等，错误的预测行为将会造成不可预计的灾难性后果

2016年5月，弗洛里达一条高速公路上，发生了全球首例自动驾驶死亡事件

2021年8月，沈海高速涵，启用NOP领航状态的蔚来ES8追尾了内侧道路的高速公路养护车辆

- 对深度学习系统进行测试，是深度学习系统开发环节中十分重要的一环

背景介绍

- 系统测试
  - 能否像传统软件系统一样测试深度学习系统?
  - 如何测试深度学习系统?

历史发展

历史发展

- 发展阶段
  - 第一阶段，不可测试 阶段
    - 深度学习系统诞生 至 2007年
  - 第二阶段，黑盒测试 阶段
    - 2007年 至 2017年
  - 第三阶段，覆盖测试 阶段（白盒测试阶段）
    - 2017年 至 2019年
  - 第四阶段，各种技术百花齐放
    - 2019年 至 今

历史发展

- 不可测试阶段
  - 样本集划分
    - 留出法：将样本集划分为互斥的集合一个作为训练集，另一个作为测试集
    - 交叉验证法：将样本集划分成 k 个大小相似的互斥子集，每次选择 k-1 个子集作为训练集，余下的的子集作为测试集
    - 自助法：每次从样本集 D 中拷贝一个样本放入另一集合 D’ ，重复若干次，将 D’ 作为训练集，将D \ D’ 作为测试集
  - 模型训练阶段完成的模型评估

— 2007

历史发展

- 不可测试阶段
  - 2007年，Murphy 等人最早对机器学习系统提出了系统测试的想法
  - 主要问题
    - 测试预言（Test Orecle）问题
    - 测试充分性（Test Adequacy）验证问题
  - 仅仅测试了该类系统是否可以正确实现规范并满足用户的期望，而没有对模型判别的准确性做出测试

— 2007

历史发展

- 测试预言
  - 在测试时确定与实际结果进行比较的预期结果的源。它可能包括现有系统（作基准）、用户手册、或个人的专业知识等，但不是代码

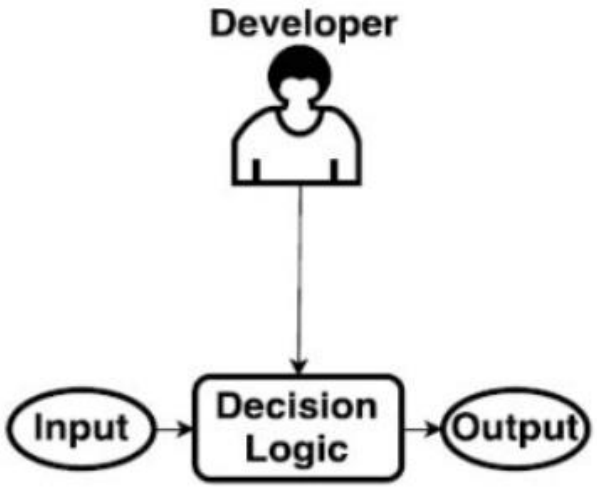
    —— ISTQB ETM
  - *A test case consists of two parts: a test input to exercise the program under test and a test oracle to check the correctness of the test execution*

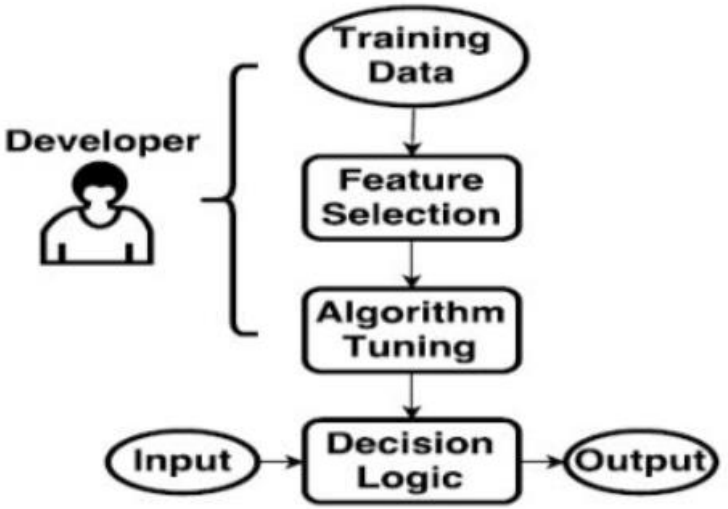
    —— 北京大学 谢涛
  - 简单来说，测试预言就是测试结果的参照物

- 为什么测试预言问题会成为深度学习系统测试的主要问题?

历史发展

- 与传统软件系统的差异
  - 传统软件系统将其逻辑表示为由人类创造的知识流
  - 而深度学习系统由数据、模型和框架复合产生
    - 通过神经元权重和非线性激活函数来表征其行为





- 深度学习系统的系统行为逻辑不受开发者直接影响，如何找到测试预言？

历史发展

- 测试充分性
  - 对于深度学习系统
    - 输入域往往是无穷无尽的
  - 对于一个测试活动
    - 测试样本基本不可能覆盖所有输入
    - 模糊测试无法直接应用于深度学习系统

- 是否可以提供一个标准，能够直观地反应测试活动是否可以全面、充分地揭示深度学习系统故障?

历史发展

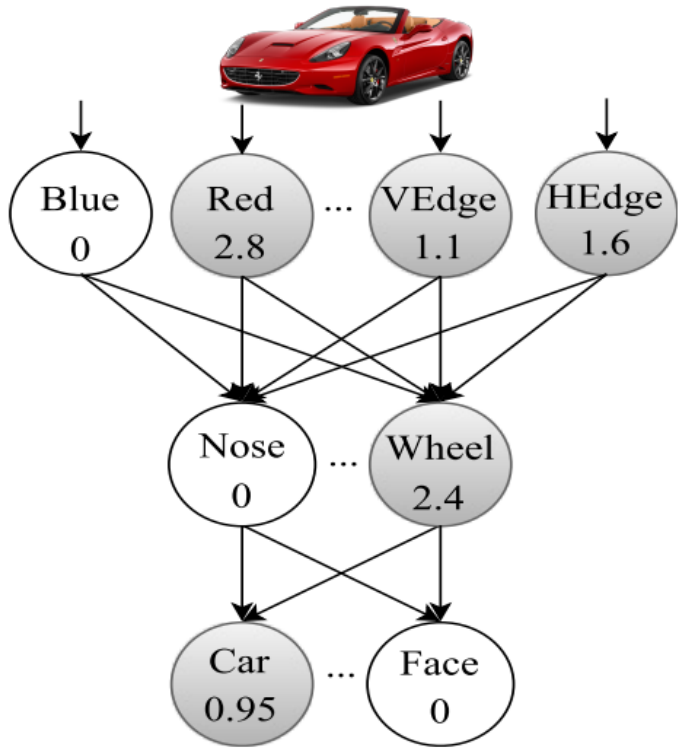
- 黑盒测试阶段
  - 之后，Murphy 等人再次对机器学习系统测试做出研究
    - 变形测试/蜕变测试来解决测试预言问题
    - 变形测试：找到一种变形关系，使得原测试样本的输出和变形后测试样本的输出满足某种特定关系，通过检测这种特定关系来测试系统
  - 尽管他们并未进行评估实验，但这项工作确定了变形测试可应用于机器学习系统

……

— 2007 | 2007 — 2017

历史发展

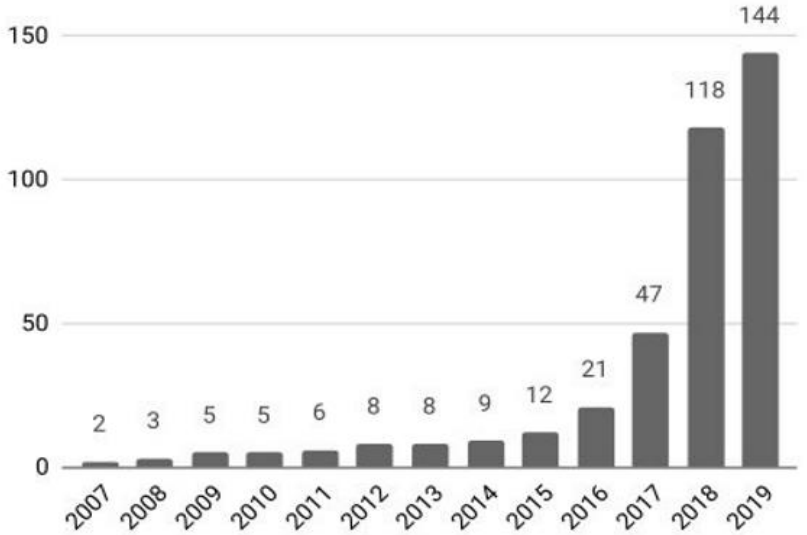
- 覆盖测试阶段
  - 2017年，Pei 等人首次提出一种针对深度学习系统的白盒测试方法
    - 首次引入覆盖测试，提出神经元覆盖率，用于解决测试充分性验证问题
      - 神经元覆盖率：给定样本输入下，激活神经元数量与总量之比
      - 对后续各类神经元覆盖率的提出奠定基础
    - 引入差分测试，用于解决测试预言问题
      - 差分测试：对于同一样本，功能相似的模型应有相同输出
      - 对后续突变测试引入深度学习系统测试奠定基础
    - 通过联合优化，生成测试样本
      - 最大化神经元覆盖率，最大化输出差异



— 2007 | 2007 — 2017 | 2017 — 2019

历史发展

- 覆盖测试阶段
  - 该方法提出后，深度学习系统测试的相关研究呈现出爆发式增长
  - 大多研究者参考神经元覆盖率提出各种覆盖
    - DeepGauge — K 节神经元覆盖、强神经元覆盖等
    - DeepCT — t-way 组合覆盖
    - DeepCover — SSC 覆盖
    - ……
  - 通过最大化各类神经元覆盖率，引导生成测试样本



— 2007 | 2007 — 2017 | 2017 — 2019

历史发展

- 现阶段
  - 从2019年开始，该领域百花齐放
  - 测试方法上
    - 各类传统软件系统的测试方法被不断引入（变形测试、覆盖测试、模糊测试、符号执行、突变测试等等）
  - 测试对象上
    - 不再局限于模型，针对框架、样本数据的测试不断出现
  - 测试属性上
    - 不在局限于模型的安全性，针对模型公平性、隐私性、效率的测试不断发展

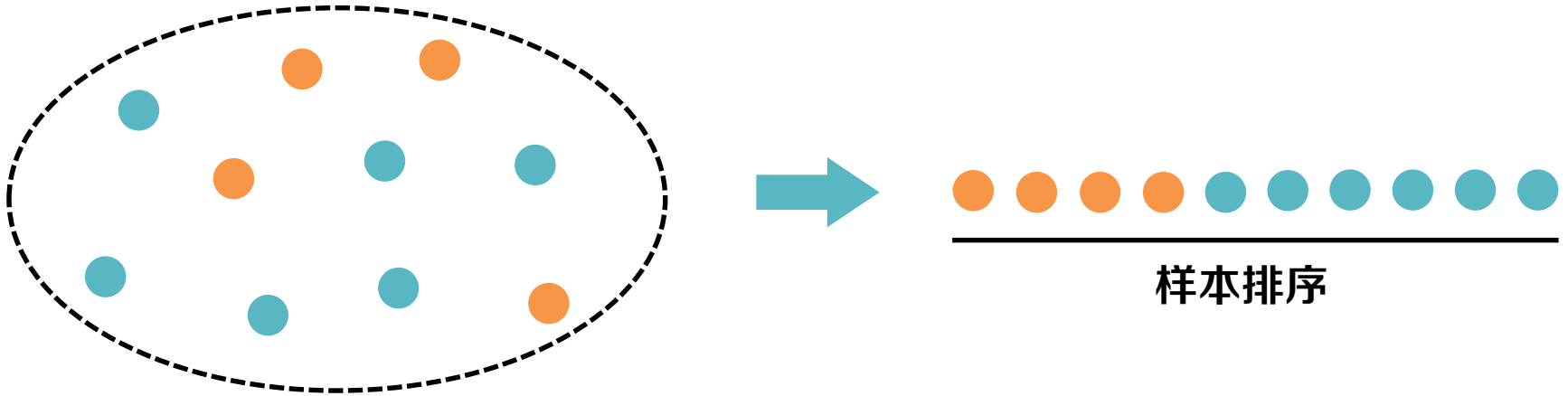
历史发展

- 现阶段
  - 测试目标上，不在局限于
    - 测试预言问题的解决
    - 测试充分性的验证
    - 测试样本的生成
  - 逐渐衍生出
    - 测试样本优先级排序及测试样本的选择

基本概念

基本概念

- 测试样本优先级排序
  - 对于一组无标签样本
    - 人力资源有限，无法为所有样本进行手工标注
    - 选择哪些样本进行标注，才能尽可能找到模型的问题?
  - 目标：在无标签样本中找到引发模型预测错误的样本



表示使模型预测错误样本，表示使模型预测正确样本

算法原理

算法原理

T	在无标签样本中找到引发模型预测错误的样本
I	待检测模型、待检测模型的训练集、无标签测试样本集
P	1.计算训练样本和测试样本的激活迹 2.计算LSA或DSA度量 3.根据度量进行排序
O	测试样本排序

P	覆盖测试中丢失了神经元输出连续性的特征
C	算法运行时间过长
D	定义LSA和DSA度量
L	ICSE 2019

算法原理

- 惊喜充分性（Surprise Adequacy）
  - 提出两个假设
    - 假设1：如果两个输入是相似的，那么它们输入深度学习系统后的输出也应该是相似的
    - 假设2：测试集样本越多样化，对深度学习系统的测试就越有效
  - 当前的覆盖标准的问题
    - 仅仅计算神经元的输出是否满足神经元的某种特定条件
    - 所反映单个输入的信息量较少
    - 没有利用神经元输出是一个连续变量的特点

算法原理

- 惊喜充分性（Surprise Adequacy）
  - 用于衡量在当前训练样本下深度学习系统对单个输入样本的惊喜程度（Surprise）
  - 惊喜程度可以用
    - DL系统在训练过程中遇到相似输入的可能性
    - 给定测试样本和训练数据的某种距离
  - 激活迹（Activation Trace）
    - 令N $= \{n_1, n_2, \ldots\ldots\}$表示神经网络$D$的一组神经元，X $= \{x_1, x_2, \ldots\ldots\}$为一组输入
    - 使用$\alpha_n(x)$表示神经元n对于输入x的输出
    - 使用$\alpha_{\mathrm{N}}(x)$表示一组神经元N对于输入x的输出向量，称$\alpha_{\mathrm{N}}(x)$为x在N上的激活迹
    - 使用$\mathrm{A_N}(X) = \{\alpha_{\mathrm{N}}(x) | x \in X\}$表示$X$在$N$上的一组激活迹

算法原理

- 基于可能性的惊喜充分性（LSA）
  - 核心思想
    - 使用采用核密度估计（KDE）来获得输入数据的分布密度函数，由此密度函数估计随机变量的特定值的相对似然
  - 公式
    - 给定带宽矩阵H和高斯核函数K，对于一个新的样本输入x在训练集T上的密度估计

$$\hat{f}(x) = \frac{1}{\left|A_{N_L}(T)\right|} \sum_{x_i \in T} K_{H\left(\alpha_{N_L}(x) - \alpha_{N_L}(x_i)\right)}$$

    - 基于此，定义LSA度量（概率密度降低时增加）

$$LSA(x) = -log\left(\hat{f}(x)\right)$$

算法原理

- 基于距离的惊喜充分性（DSA）
  - 核心思想
    - 使用距离度量测试样本x的激活迹和训练样本的激活迹
    - 更接近类边界的输入在测试输入多样性方面更惊喜和有价值的
  - 公式表示
    - 对于一个测试样本x，对它的预测类别$c_x \in C$，定义参考点x_a为相同类中最接近x的点，则x与x_a之间的距离表示为

$$dist_a = \|\alpha_n(x) - \alpha_n(x_a)\|$$

    - 找到类别不同于c_x且与x_a距离最近的参考点x_b，同理，x_a与x_b之间的距离表示为

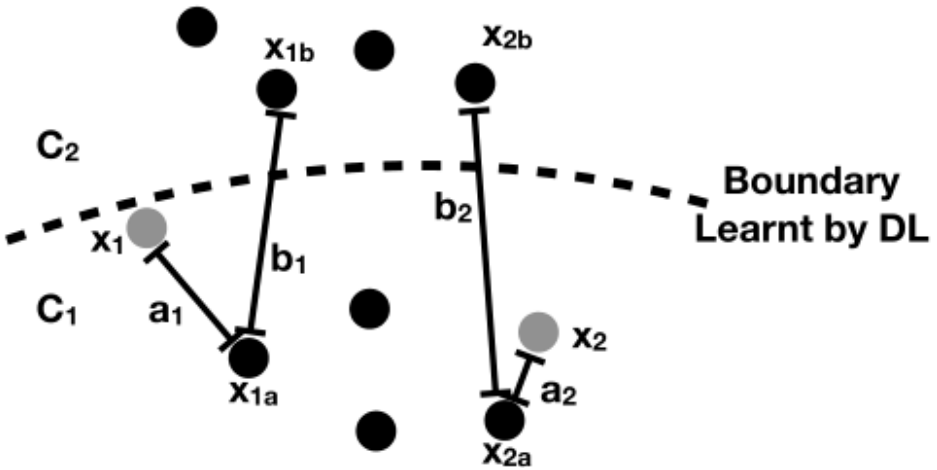
$$dist_b = \|\alpha_n(x_a) - \alpha_n(x_b)\|$$

算法原理

- 基于距离的惊喜充分性（DSA）
  - 公示表示
    - 基于此，定义DSA度量

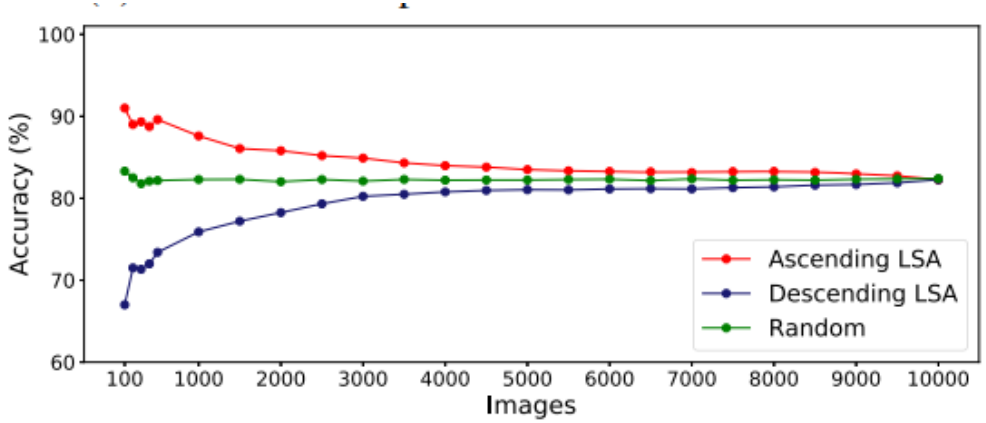
$$DSA(x) = \frac{dist_a}{dist_b}$$

  - DSA度量就是在比较x到与它相同类的训练样本激活迹的距离和该参考样本到不同类的训练样本激活迹的距离

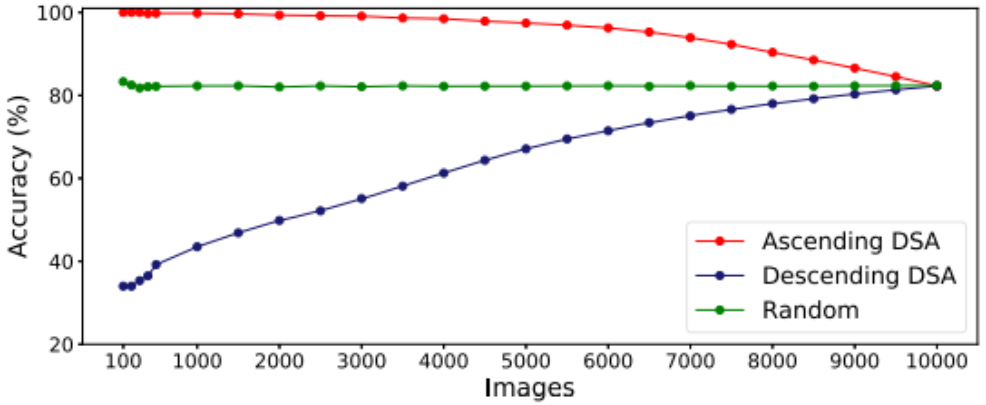


算法原理

- 算法实验
  - 计算每个测试样本的LSA和DSA，按照从大到小、从小到大和随机的方式进行测试，观察平均准确率的变化



(c) Selected test inputs based on LSA in CIFAR-10

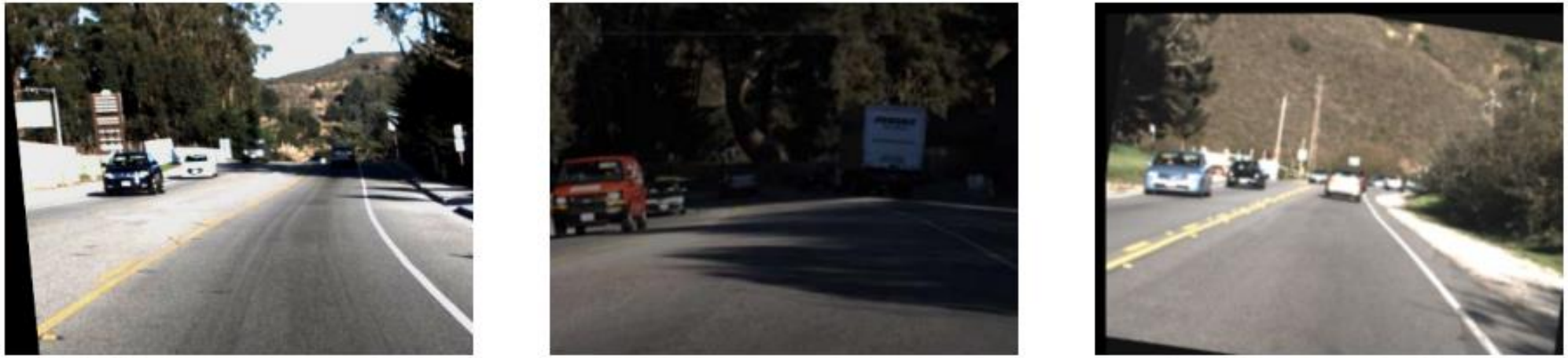


(d) Selected test inputs based on DSA in CIFAR-10

  - 结果：由SA从大到小进行测试，平均准确率不断提升；反之，则逐渐下降；随机选择则介于两者之间。

算法原理

- 算法实验
  - 作者挑选了高、中、低LSA的三组图像进行对比
  - 比较不同LSA的图像肉眼识别程度



(a) Low LSA

算法原理

- 算法实验
  - 作者挑选了高、中、低LSA的三组图像进行对比
  - 比较不同LSA的图像肉眼识别程度

(b) Medium LSA

算法原理

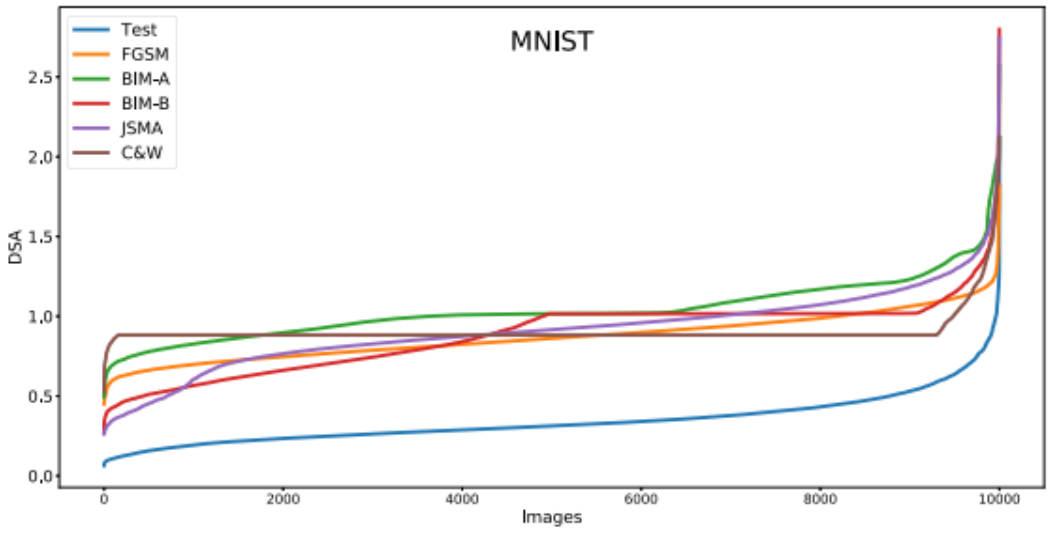
- 算法实验
  - 作者挑选了高、中、低LSA的三组图像进行对比
  - 比较不同LSA的图像肉眼识别程度

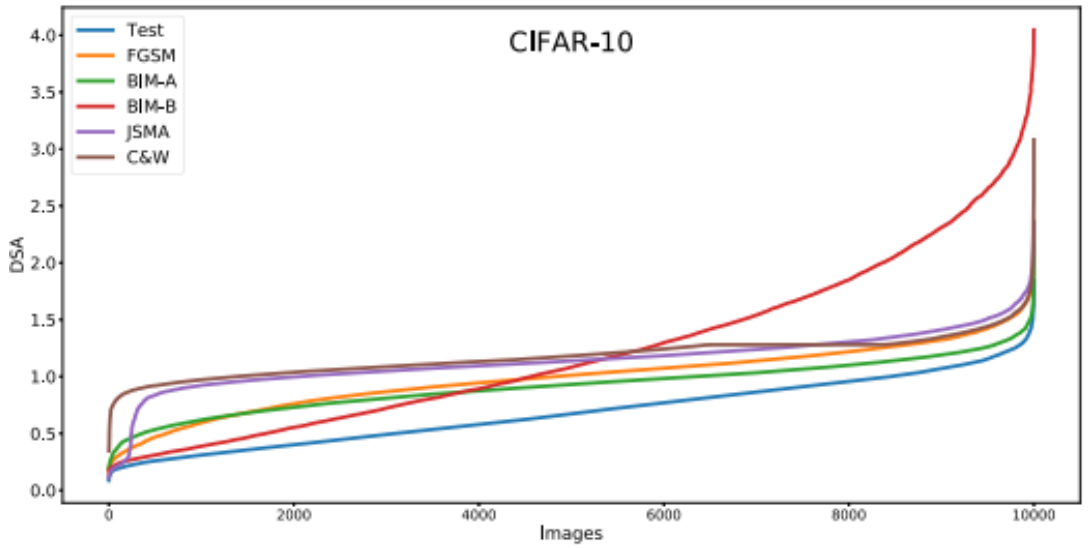
(c) High LSA

  - 结果：LSA越高，肉眼识别难度越大

算法原理

- 算法实验
  - 使用不同的对抗样本生成技术（FGSM、BIM-A、BIM-B、JSMA和C&W）生成多个对抗样本集
  - 比较它们和原始样本的DSA





  - 结果：五种对抗样本的DSA明显高于与原始样本的DSA

算法原理

- 实验结论
  - SA方法是有效的，它可以衡量在当前训练样本下深度学习系统对单个输入样本的惊喜程度
  - 具有较高SA的输入更难被正确分类，无论是人还是深度学习系统
  - 对抗样本的SA更高，可以根据这一点检测对抗样本

- 缺点
  - 计算时间较长
    - LSA和DSA方法的时间复杂度均为O(mn)
    - 其中LSA因为使用了stats.gaussian_kde()函数，运行时间巨长
  - DSA无法应用于回归任务

算法原理

T	在无标签样本中找到引发模型预测错误的样本
I	待检测模型、待检测模型的训练集、无标签测试样本集
P	1.将无标签测试样本输入待检测模型，获取预测概率矩阵 2.根据预测概率矩阵计算Gini系数 3.根据Gini系数进行排序
O	测试样本排序

P	SA方法运行时间过长
C	无法应用于回归任务
D	通过计算Gini系数进行排序
L	ISSTA 2020

算法原理

- **DeepGini**
  - 核心思想
    - 基尼系数越大，说明样本越靠近分类边界，深度学习系统对样本的不确定性也越高
  - 举例
    - 对于一个二分类模型
      - 样本A的输出概率矩阵 [0.99, 0.01]
      - 样本B的输出概率矩阵 [0.51, 0.49]
    - 那么从直观上来看，样本B比样本A更靠近分类边界，那么它出错的可能性也越大
  - 公式

$$gini(x) = 1 - \sum_{i}^{n} (p_i)^2$$

算法原理

- 算法实验
  - 使用DeepGini和其他基于覆盖测试的方法对测试样本进行排序
    - 比较最终的排序结果
    - 比较运行时长
  - 基于覆盖测试的方法如何对样本进行排序?
  - CAM
    - 每次选择覆盖率最高的测试，以此类推
  - CTM
    - 根据之前选择的反馈来选择下一测试样本
    - 选择使当前覆盖率最高的

Test	Program Statement							
	1	2	3	4	5	6	7	8
A	X	X	X			X	X	X
B	X	X	X				X	X
C	X	X	X	X				
D					X	X	X	X

算法原理

- 算法实验
  - 如何衡量排序结果的好坏?

$$APFD = 1 - \frac{\sum_{i=1}^{k} o_i}{kn} + \frac{1}{2n}$$

  - 其中
    - n代表测试样本总数
    - k代表错误分类样本总数
    - o_i代表第i个错误分类样本的排序

算法原理

- 算法实验

Metrics	Param.	Original Tests						Original Tests + Adv					
		Max Cov.		CTM		CAM		Max Cov.		CTM		CAM	
		%	#	Time (s)	APFD	Time (s)	APFD	%	#	Time (s)	APFD	Time (s)	APFD
NAC	0	100	1	2	0.638	2	0.638	100	1	11	0.340	11	0.340
	0.75	84	22	2	0.385	4	0.384	86	21	11	0.307	16	0.307
NBC	0	8	38	3	0.638	9	0.638	15	56	14	0.339	40	0.339
	0.5	0.97	5	2	0.638	5	0.637	3	11	13	0.400	20	0.400
	1	0.39	3	3	0.638	6	0.637	2	7	13	0.400	20	0.400
SNAC	0	14	35	3	0.639	9	0.639	22	48	13	0.340	31	0.340
	0.5	2	5	3	0.639	7	0.639	7	11	13	0.340	22	0.340
	1	0.78	3	3	0.638	8	0.638	4	7	13	0.340	20	0.340
TKNC	1	66	86	N/A	N/A	11	0.023	74	96	N/A	N/A	59	0.001
	2	73	67	N/A	N/A	10	0.023	79	70	N/A	N/A	48	0.001
	3	76	57	N/A	N/A	10	0.023	81	55	N/A	N/A	43	0.001
LSC	(1000, 100)	22	220	N/A	N/A	9	0.658	98	982	N/A	N/A	36	0.503
DSC	(1000, 2)	53	531	N/A	N/A	1177	0.658	97	974	N/A	N/A	4738	0.490
KMNC	1000	63	8814	N/A	N/A	34045	0.599	T/O	T/O	N/A	N/A	T/O	T/O
	10000	T/O	T/O	N/A	N/A	T/O	T/O	T/O	T/O	N/A	N/A	T/O	T/O
DeepGini	N/A	N/A	N/A	N/A	N/A	**0.45**	**0.984**	N/A	N/A	N/A	N/A	**2**	**0.991**

算法原理

- 实验结论
  - Deepgini方法比其他覆盖测试方法
    - 排序效果更好
    - 运行时间更短

- 缺点
  - 无法应用于回归任务
  - 同其他测试优先级排序方法比效果较差
    - 实验仅证明了Deepgini优于其他基于覆盖测试的方法
    - 这些方法并未被证明可应用于测试优先级排序任务

算法原理

T	在无标签样本中找到引发模型预测错误的样本
I	待检测模型、待检测模型的训练集、无标签测试样本集
P	1.对模型添加突变算子，生成若干变异模型 2.将样本输入待检测模型和变异模型中，获取输出结果 3.提取差异特征，构建XGBRanker排序器 4.对测试样本集进行排序
O	测试样本排序

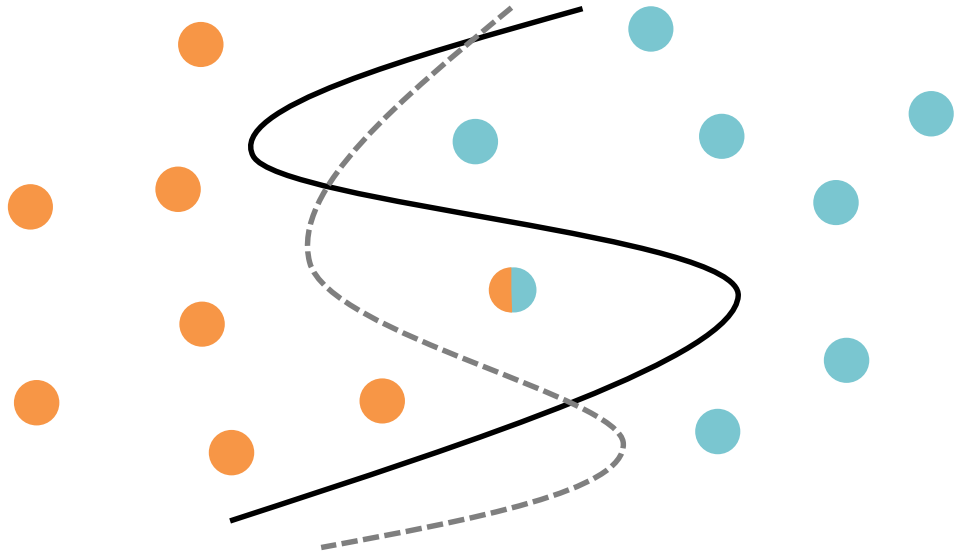
P	Gini方法无法应用于回归任务
C	需要一组有标签样本用于XGBRanker的训练
D	通过突变模型和原始模型输出的差异特征对样本进行排序
L	ICSE 2021

算法原理

- 突变测试
  - 突变测试通常对程序的源代码或者目标代码做小的改动，并把截然不同的错误行为作为预期。如果测试数据没有觉察到这种小改动带来的错误，就说明这个测试是有问题的。
  - *A testing methodology in which two or more program mutations are executed using the same test cases to evaluate the ability of the test cases to detect differences in the mutations.* ——**IEEE (Std 610.12-1990)**
- 突变测试理论的两个基本假设
  - 程序员能力假设（Competent Programmer Hypothesis，CPH）
  - 耦合效应假设（Coupling Effect，CE）

算法原理

- **Prima**
  - 核心思想
    - 样本越靠近分类边界，越有可能引发模型错误
    - 突变测试，通过引入突变引发分类边界的变动，将分类边界变动前后有差异的样本筛选出来



算法原理

- **Prima**
  - 模型突变
    - 高斯模糊：向模型权重随机添加高斯噪声
    - 权重混乱：随机选取权重，进行混序
    - 神经元逆转：随机选取神经元，将输出进行取反
    - 神经元阻断：随机选取神经元，将输出置0
  - 图片样本突变
    - 像素高斯模糊：对图片像素随机添加高斯噪声
    - 像素反色：随机选取像素，进行反色
    - 添加白色像素：随机选取像素，变为白色
    - 添加黑色像素：随机选取像素，变为黑色

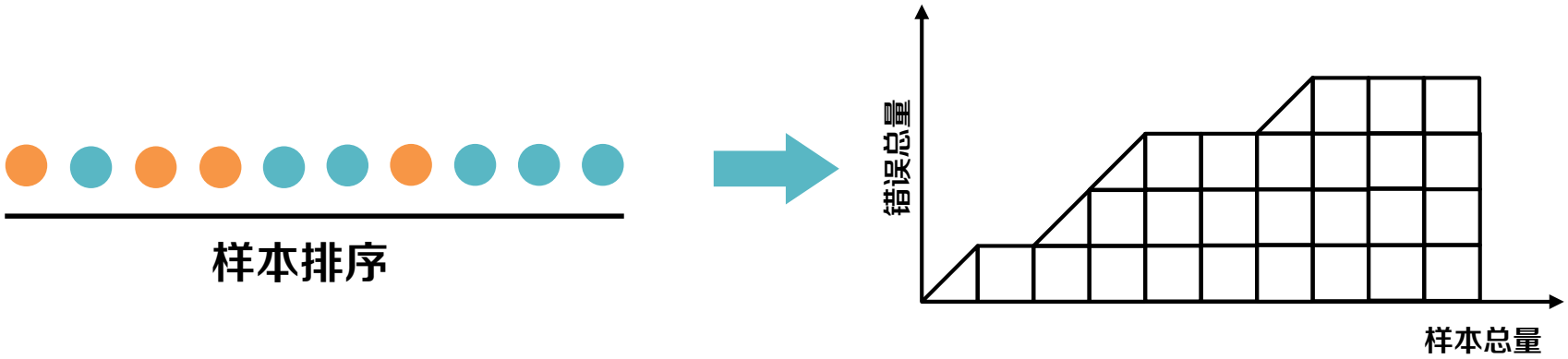
……

算法原理

- **Prima**
  - 每种突变各生成若干突变模型/突变样本
    - 突变模型：将原始样本输入到突变模型中，获取一组输出
    - 突变样本：将突变样本输入到原始模型中，获取一组输出
  - 提取每种突变特征
    - 分类任务
      - 与原预测概率矩阵的差异性：曼哈顿距离、欧式距离、切比雪夫距离、余弦距离
      - 与原预测结果的差异性：不一致总量、与不一致数量最多标签对应数量
      - 其他：余弦距离分布
    - 回归任务
      - 差值、分布差异
  - 演变成二分类任务，构建模型排序器

算法原理

- **Prima**
  - 提出 RAUC – N 用于评估排序结果的好坏
    - 横轴代表测试样本选取总量
    - 纵轴代表
      - 分类任务：选取的测试样本中，使模型出现分类错误的总量
      - 回归任务：与真实值之间的差值



- 对于前N个样本，实际排序面积与理想排序状态下面积之比

算法原理

- 算法实验
  - 在12种数据集的各种情况（增添对抗样本、更换模型等）下，共计36组，进行优先级排序

	Approach	#Best cases in RAUC- 100	200	300	500	All	Average RAUC- 100	200	300	500	All	Improvement of PRIMA (%) in RAUC- 100	200	300	500	All
C	DeepGini	2	2	2	1	3	0.751	0.753	0.752	0.755	0.847	18.24	16.47	15.69	14.97	8.50
	LSA	0	0	0	0	0	0.568	0.558	0.559	0.571	0.685	56.34	57.17	55.64	52.01	34.16
	DSA	0	0	0	0	0	0.648	0.632	0.625	0.619	0.722	37.04	38.77	39.20	40.23	27.29
	PRIMA	28	28	28	29	27	0.888	0.877	0.870	0.868	0.919	-	-	-	-	-
R	LSA	0	0	0	0	0	0.345	0.357	0.368	0.394	0.689	131.01	122.97	114.67	97.72	17.27
	PRIMA	6	6	6	6	6	0.797	0.796	0.790	0.779	0.808	-	-	-	-	-

  - 结果
    - prima的排序效果远高于其他几种
    - 在运行时间上仅低于DSA方法

Approach	Mean	Std.	Min.	Max.
DeepGini	0.1	0.1	< 0.1	1.4
LSA	1.5	1.0	< 0.1	2.8
DSA	23.5	110.1	< 0.1	616.2
PRIMA	10.4	6.2	2.4	27.7

算法原理

- 实验结论
  - Prima方法比其他几种排序方法（DSA、LSA、DeepGini）
    - 排序效果更好
    - 适用于多种场景

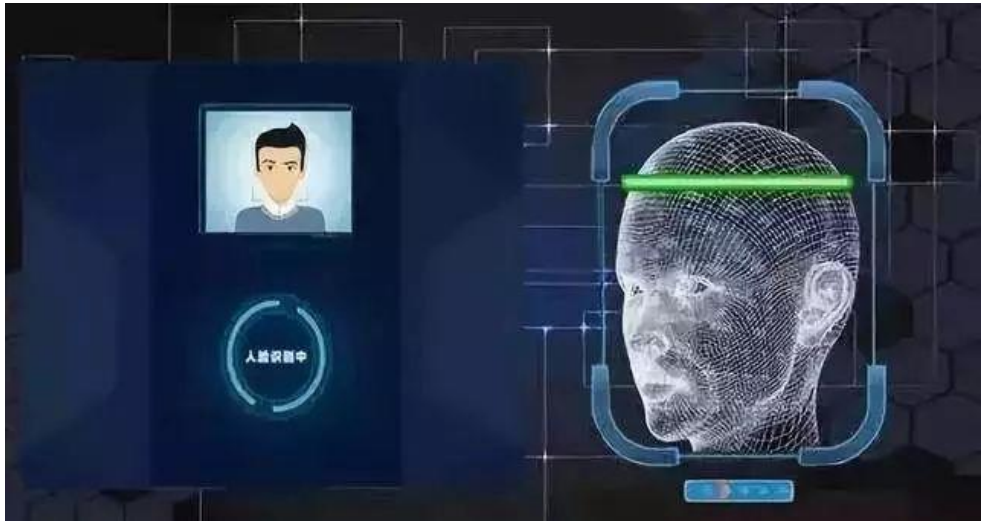
- 缺点
  - 需要一组有标签样本用于排序器的构建
    - 实验中用于构建排序器的数据是从测试样本中划分的
  - 需要原始训练集
  - 实际运行条件严苛，对内存要求非常大

应用总结

- 深度学习系统安全性测试
  - 无人驾驶
  - 恶意流量检测
  - 人脸识别

    ……

- 测试样本优先级排序
  - 资源有限条件下的各类测试活动



参考文献

- [1] Kim J, Feldt R, Yoo S. Guiding deep learning system testing using surprise adequacy[C]//2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE). IEEE, 2019: 1039-1049.

- [2] Feng Y, Shi Q, Gao X, et al. Deepgini: prioritizing massive tests to enhance the robustness of deep neural networks[C]//Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis. 2020: 177-188.

- [3] Wang Z, You H, Chen J, et al. Prioritizing Test Inputs for Deep Neural Networks via Mutation Analysis[C]//2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE). IEEE, 2021: 397-409.

知人者智，自知者明。

胜人者有力，自胜者强。

知足者富，强行者有志。

不失其所者久，死而不亡者，寿。

谢谢！