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

AI测试：历史与发展

硕士研究生 王若辉

2022 年 8 月 14 日

内容提要

- 背景简介
- 历史发展
- 应用总结
- 参考文献

内容提要

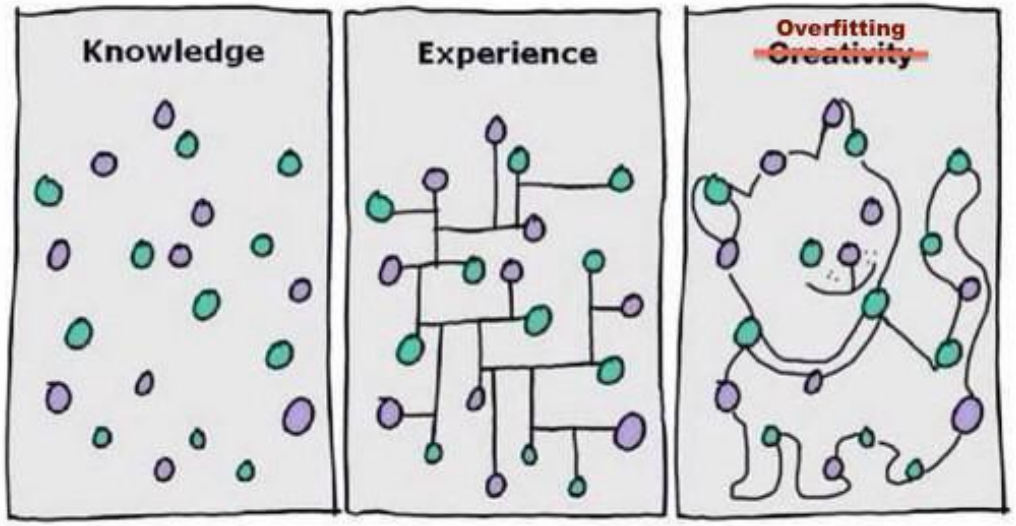
- 预期收获
  - **1.了解AI测试的发展历史**
  - **2.掌握AI测试中4种常用测试技术**
  - **3.明确实验室AI测试方向的研究空白**

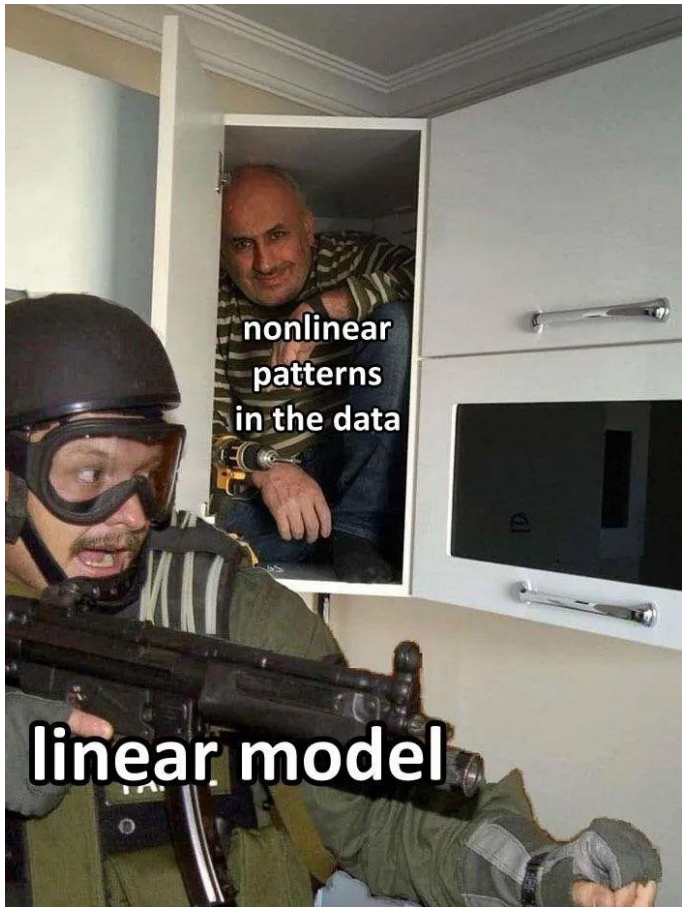
背景介绍

- 自人工智能被提出，已经过60多年积累与演进，在技术驱动和市场带动下，取得了历史性进步
- 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被普遍应用于各类系统

背景介绍

- 这在推动人工智能技术发展的同时也引起了人们对人工智能可靠性的关注。
  - 人工智能技术自身存在不少的缺陷
    - 模型的过拟合、欠拟合
    - 数据不平衡、数据污染
    - 框架或组件自身存在的漏洞





背景介绍

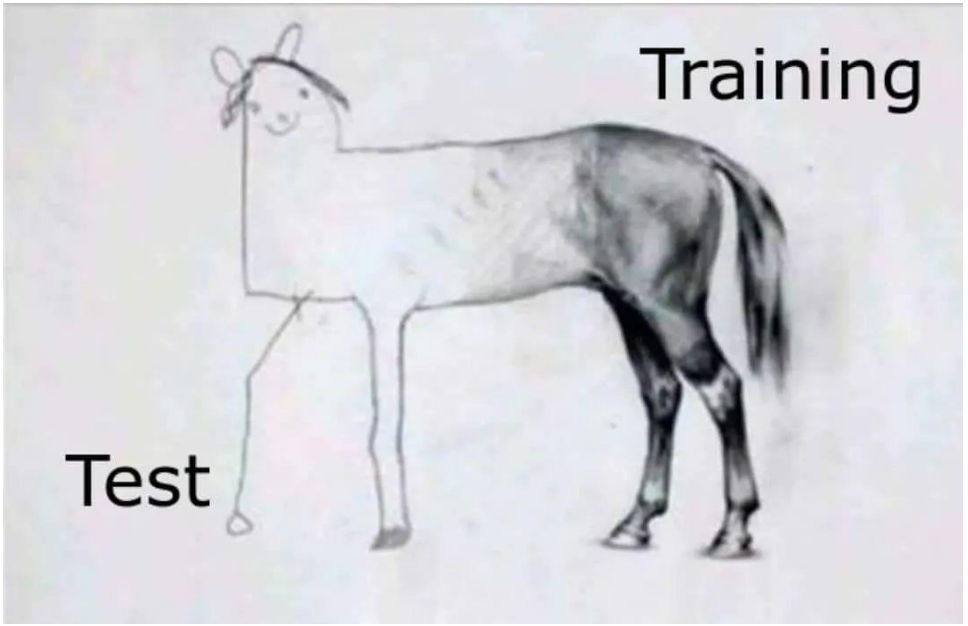
- 这些缺陷将导致人工智能系统在正确性、鲁棒性、隐私性、公平性和效率等各方面出现严重的问题。
  - 特别是在对安全至关重要的领域，如交通、医疗卫生、制造领域
  - 将直接对人的生命财产安全造成严重威胁

2022年5月，SpiceJet旗下一架波音737自动驾驶系统失灵，造成17人重伤

2022年8月，一辆开启了LCC车道居中辅助的小鹏P7在宁波高架行驶过程中与前方一辆检查故障的车辆相撞

背景介绍

- 软件测试
  - ***Software testing refers to any activity that aims to detect the differences between existing and required behavior.***(检测现有行为和所需行为之间差异)
  - 对人工智能系统进行测试，是人工智能系统开发环节中十分重要的一环



背景介绍

- 人工智能系统测试 - **Test For AI**
  - 测试属性 - **Why to test?**
  - 测试组件 - **What/Where to test?**
  - 测试流程 - **How to test?**

历史发展

著名学者 Murphy

2007/06/21

人工智能系统是不可测试的！！！

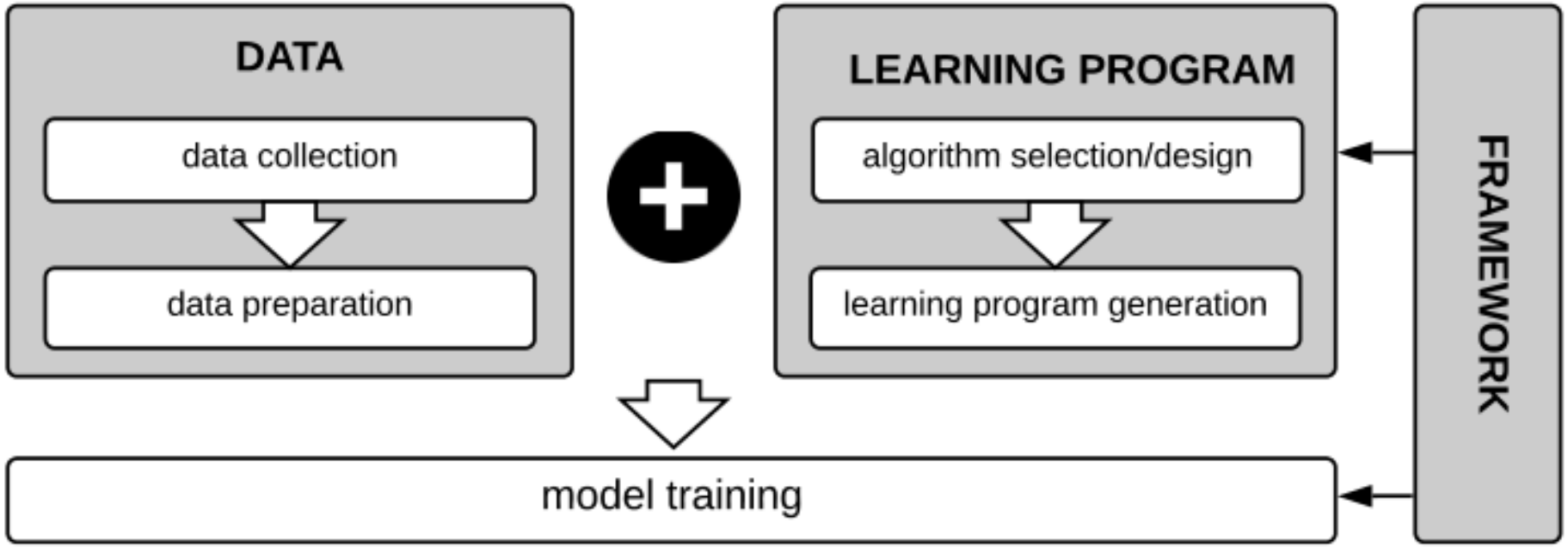
10

历史发展

- 问题
  - 为什么Murphy认为人工智能系统是不可测试的?
    - **A.** 人工智能系统缺乏可解释性
    - **B.** 人工智能系统与传统软件系统存在巨大差异
    - **C.** 人工智能系统缺陷尚不明确
    - **D.** 钝角

历史发展

- 不同的产生方式
  - 传统软件系统将其逻辑表示为由开发者创造的知识流
  - 而人工智能系统由数据、模型和框架复合，通过训练产生



历史发展

- 输出结果
  - 人工智能系统并非由开发者逻辑指导产生输出结果
  - 人工智能系统的输出结果是不确定的

```
In [2]:  # 加载模型
         CatDogModel = load_model("王某人训练的猫狗识别模型.h5")
         # 识别输入
         CatDogModel.predict(这玩意)
```

([狗:0.88][猫:0.12])

```
In [3]:  if Looklike(这玩意) == 猫:
             print("这是猫！")
         else:
             print("这是狗！")
```

这是狗！

- 这种不确定性将导致人工智能系统的测试活动无法找到合适的测试预言

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

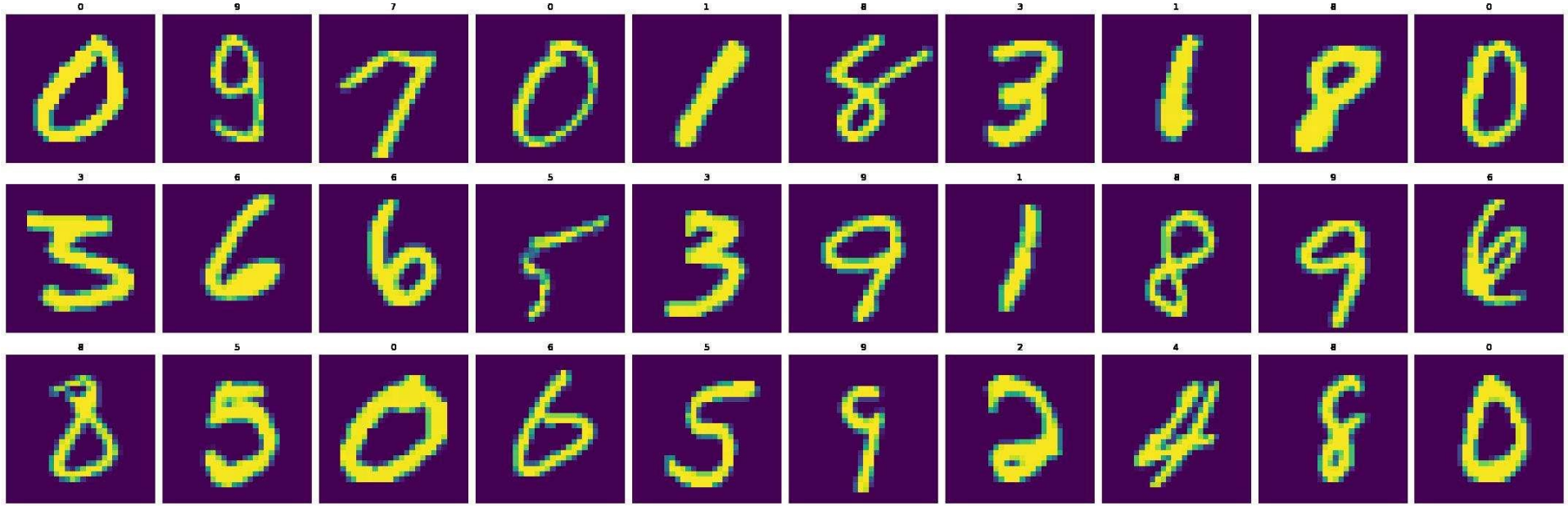
    —— **ISTQB ETM**
  - ***A test case consists of two parts: a test input to exercise the program under test and a test oracle to check the correctness of the test execution***

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

- 有没有解决测试预言问题的简单方法?

历史发展

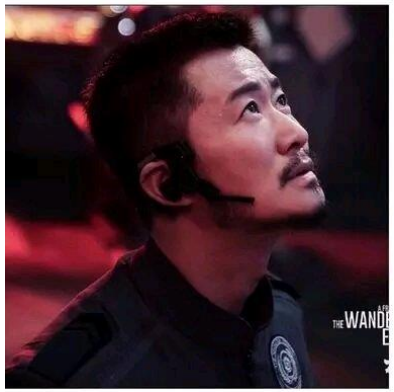
- 手工标记
  - 领域专家经验，对样本进行人工打标签的方式



- 手工标记存在什么样的缺点?

历史发展

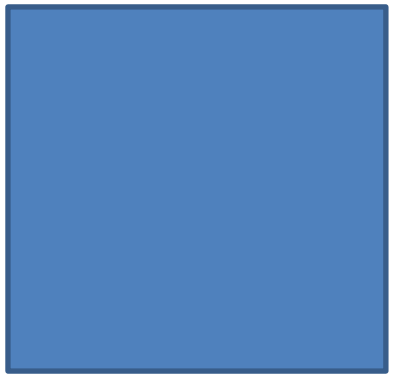
- 手工标记的缺点
  - 成本高：手工标记的时间成本、人力成本
  - 易出错：手工标记容易产生错误标记
  - 专家经验：复杂领域需要标记者具备领域专家经验



请标记出下图中的吴京、茄子和荒木吕飞彦

历史发展

- 测试输入
  - 人工智能系统的测试输入是无穷无尽的
    - 无限的输入空间



    - 缺乏测试充分性标准

  - 传统软件测试使用测试覆盖作为测试充分性的评估标准
    - 语句覆盖、判定覆盖、条件覆盖等

历史发展

著名学者 Murphy

...

2007/12/13

人工智能系统是可以测试的！！！

历史发展

著名学者 Murphy ...

不过只能测一点点。。

测测系统是否可以正确实现规范并满足用户的期望

著名学者 Murphy ...

“著名学者 Murphy” 撤回了一条消息

“著名学者 Murphy ” 撤回了一条消息

我 拍了拍 “著名学者 Murphy”并献上了自己的膝盖

算法原理

T	系统测试 - 变形测试
I	一组测试样本（用例）及该组测试样本的预期结果
P	1.构建测试样本变形前后的确定性关系 2.将变形前后的测试样本输入到待测试系统中 3.验证输出是否满足确定性关系
O	系统的缺陷行为

历史发展

- 情景问题
  - 小明写了这样一段函数

```
import math

def test_sin(x):
    return math.tan(x)*math.cos(x)
```

  - 在我们没有任何工具的前提下，如何测试这段程序?

```
In [15]: test_sin(1)
Out[15]: 0.8414709848078966
```

历史发展

- **2008年**
  - **Murphy** 等人对机器学习系统测试做出研究，提出变形测试/蜕变测试来解决测试预言问题，并首次提出了六种变形关系
    - 加法：在数值上增加一个常数
    - 乘法：将数值乘以常数
    - 乱序：重新排列输入顺序
    - 反转：反转输入顺序
    - 包含：移除训练数据的一部分
    - 排他：添加附加数据

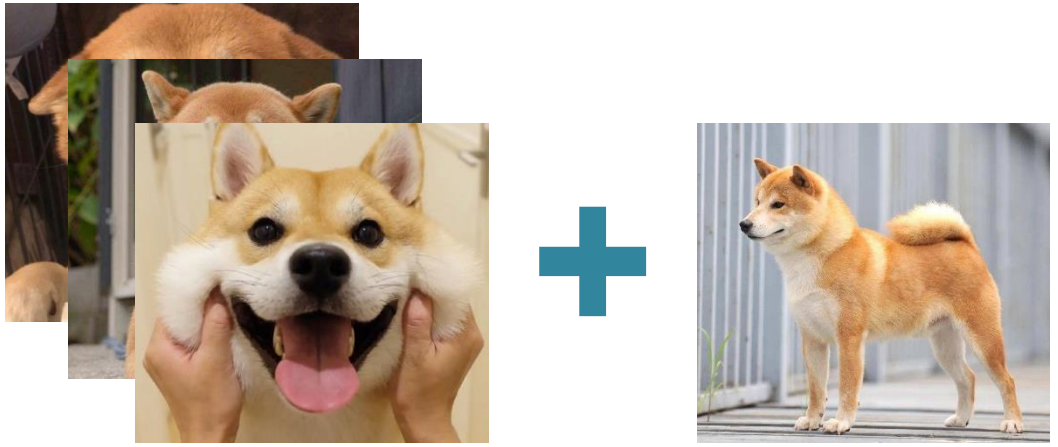
  - 尽管他们并未进行评估实验，但这项工作确定了变形测试可应用于人工智能系统
  - 也正式揭开了智能化系统测试的序幕

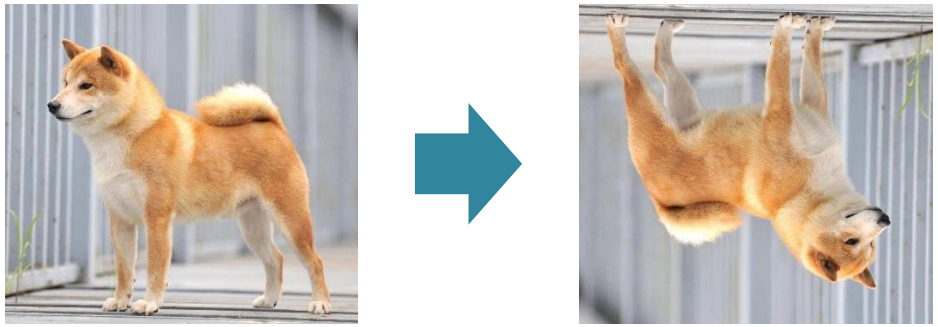
历史发展

- 根据变形关系是否执行对每个具体实例的修改，变形关系可被分为粗粒度变形关系和细粒度变形关系
  - 粗粒度：
    - 扩大数据集
    - 更改特征顺序

    ……
  - 细粒度：
    - 改变样本特征
    - 改变样本标签

    ……





- 变形关系不一定是等式
  - **Zhang**等人通过在训练数据中注入噪声，在噪声增加时检查训练精度的下降率

历史发展

- 情景问题
  - 小明写了这样一段函数

```
import math

def test_sin(x):
    return math.tan(x)*math.cos(x)
```

- 除了变形测试外，还有什么可以解决测试预言问题的方法?
  - 程序输出是一个不确定的结果
  - 如何将不确定结果转变为确定结果
    - “等比数列求和？”

历史发展

- 情景问题
  - 小明写了这样一段函数

```
import math

def test_sin(x):
    return math.tan(x)*math.cos(x)
```

  - 小明找大明写了这样一段功能相同的函数

```
import math

def test_Sin(x):
    return math.sin(x)
```

历史发展

- 情景问题
  - 经过对比，发现

```
In [3]: test_sin(1)
Out[3]: 0.8414709848078966

In [4]: test_Sin(1)
Out[4]: 0.8414709848078965

In [5]: test_sin(math.pi/6)
Out[5]: 0.5

In [6]: test_Sin(math.pi/6)
Out[6]: 0.49999999999999994
```

  - 通过比较两个程序对相同输入的输出结果，发现程序的缺陷

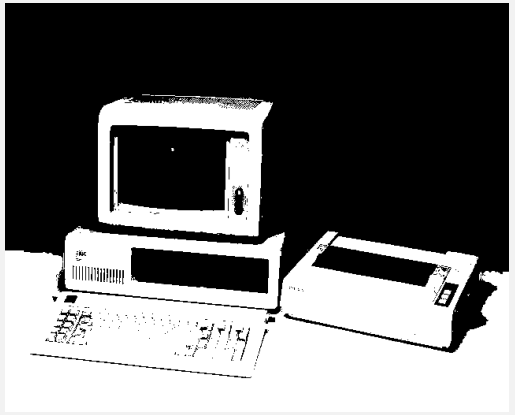
- 是否可以将这种思想用于人工智能系统的测试?

历史发展

朋友圈

知名学者 **Davis**

如果一个算法的多个实现在一个相同的输入上产生不同的输出，那么至少一个实现包含缺陷。



1981年5月

❤ 知名学者 **Pei**, 知名学者 **Alebiosu**, 知名学者 **Pham**

我：大佬 **666**！！！

算法原理

T	系统测试 - 差分测试
I	1 组测试样本（用例） 1 个以上与待测系统功能相同的软件系统
P	1. 将测试样本分别输入到待测系统和相同功能系统 2. 对比输出之间是否存在差异
O	系统的缺陷行为

历史发展

- **2017年**
  - **Pei** 等人首次将差分测试引入人工智能系统测试用于解决测试预言问题
    - 使用不同框架生成若干个功能相同的人工智能模型
    - 最大化模型间的差异行为为优化目标，生成测试样本
  - **N-Version**
    - 使用同一程序的不同版本
  - “镜像”程序
    - 基于现有训练数据使用不同算法生成程序

- **Deepxplore: Automated white box testing of deep learning systems.**
  - 不仅首次引入了差分测试，同样首次引入了覆盖测试

算法原理

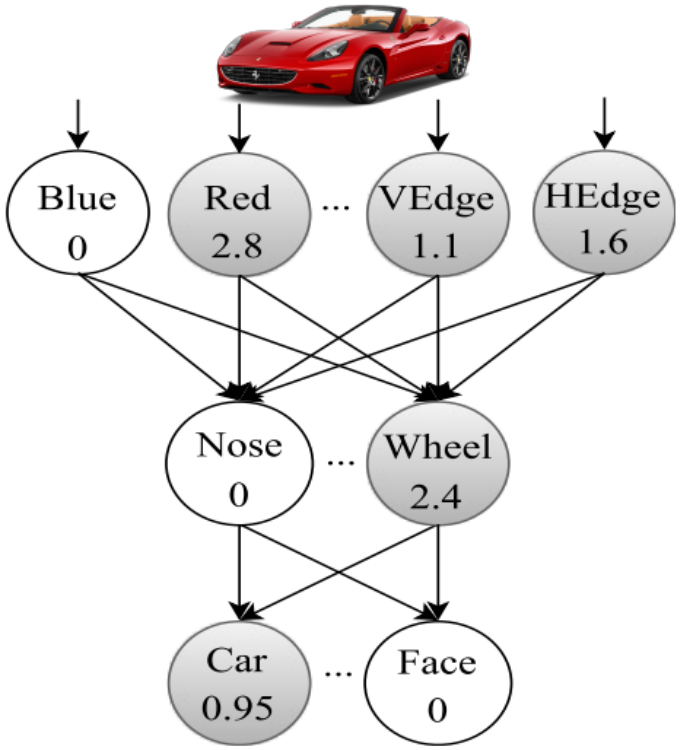
T	系统测试 - 覆盖测试
I	1 组测试样本
P	1. 随机选择测试样本对系统测试 2. 计算当前测试活动覆盖标准下的覆盖率 3.循环测试活动，直到覆盖率到达阈值
O	系统的缺陷行为

历史发展

- 覆盖测试
  - 一种白盒测试方法
    - 全面了解程序内部逻辑结构、对所有逻辑路径进行测试。
    - 从检查程序的逻辑着手，得出测试数据。
  - 传统软件系统测试
    - 语句覆盖——每条可执行语句。
    - 判定（分支）覆盖——每个判定表达式。
    - 条件覆盖——判定表达式的每个条件的值。
    - 判定/条件覆盖——每个判定表达式 U 判定表达式的每个值。
    - 多重条件覆盖——每个判定的所有可能的条件取值组合。
    - 路径覆盖——每条可能的路径。

历史发展

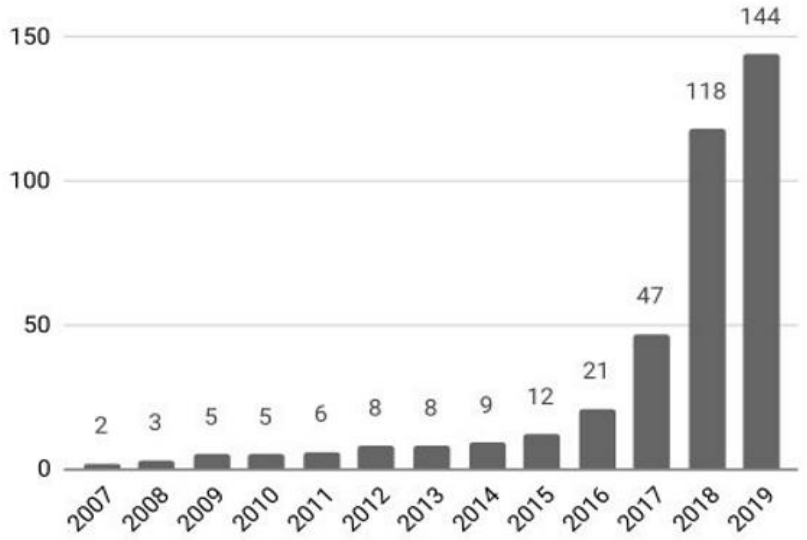
- 将覆盖测试技术应用于智能化系统
  - DeepXplore (SOSP 2017)
  - 意义：
    - 首次将白盒测试应用于智能化系统测试
    - 提出神经元覆盖率作为测试充分性验证标准
  - 神经元覆盖率：
    - 神经元输出大于设定阈值，即为激活神经元
    - 神经元覆盖率 = 激活神经元数量 / 神经元总量



- 神经元覆盖率存在哪些问题？

历史发展

- 神经元覆盖率的发展
  - 将神经元激活域细化
    - **k**节神经元覆盖率：神经元激活值**k**段细化
    - 神经元边界覆盖率：神经元激活值分布统计
    - **Top-k** 神经元覆盖率：以层为单位统计神经元激活信息
  - 引入组合关系
    - **t-way** 组合稀疏/稠密覆盖率：将组合引入
  - 扩充激活关系
    - 神经元输出覆盖率：将权重引入
    - 惊喜覆盖率：引入**SA**概念
- 智能化系统的覆盖测试是否合理?
- 是否还有其他方法作为测试充分性的验证标准?



算法原理

T	系统测试 - 突变测试
I	1 组测试样本
P	1. 对待测试系统引入突变算子生成若干突变系统 2. 将测试样本分别输入待测试系统和若干突变系统 3. 计算样本“杀死”突变数量，评估测试活动有效性
O	系统的缺陷行为

历史发展

- 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**）

历史发展

- 示例
  - 现在有一个为患者分配医生的程序
    - 1.读取病人的年龄
    - 2.如果病人大于14岁，就会分配一个普通医师作为他们的主要医生

```
1：  input Age;
2：  if Age > 14 do:
3：        Doctor = General Physician();
4：  end if;
```

  - 作为一名测试人员，给出的测试数据集是{0,13,20}
  - 目的：评估测试数据集的质量

历史发展

- 示例
  - 对程序进行以下几种突变
    - **Mutant 1：if *Age* > 14 do: => if *Age* < 14 do:**
    - **Mutant 2：if *Age* > 14 do: => if *Age* =14 do:**
    - **Mutant 3：if *Age* > 14 do: => if *Age* >=14 do:**
    - **Mutant 4： *Doctor* = General Physician(); => *Doctor* = Wang();**

    - **Mutant 5: if *Age* > 14 do: => if *Age* %%14 do:**
    - **Mutant 6: 删除 *Doctor* = General Physician();**
    - **Mutant 7: if *Age* > 14 do: => if *Age* >14 & *Age* >14 do:**

历史发展

- 示例
  - 输出结果（√表示第三行函数执行，×反之，err表示报错）

Test set-Data	Expected Result	Moutant 1	Moutant 2	Moutant 3	Moutant 4	Moutant 5	Moutant 6	Moutant 7
0	×	√	×	×	×	**err**	×	×
13	×	√	×	×	×	**err**	×	×
20	√	×	×	√	×	**err**	×	√

历史发展

- 示例
  - 输出结果（√表示第三行函数执行，×反之，err表示报错）

Test set-Data	Expected Result	Moutant 1	Moutant 2	Moutant 3	Moutant 4	Moutant 5	Moutant 6	Moutant 7
0	×	√	×	×	×	err	×	×
13	×	√	×	×	×	err	×	×
20	√	×	×	√	×	err	×	√

  - 测试数据集“杀死”了 1、2、4、5、6，未能“杀死”3、7。
  - 如果测试数据集“杀死”了所有突变体，它就是有效的。

历史发展

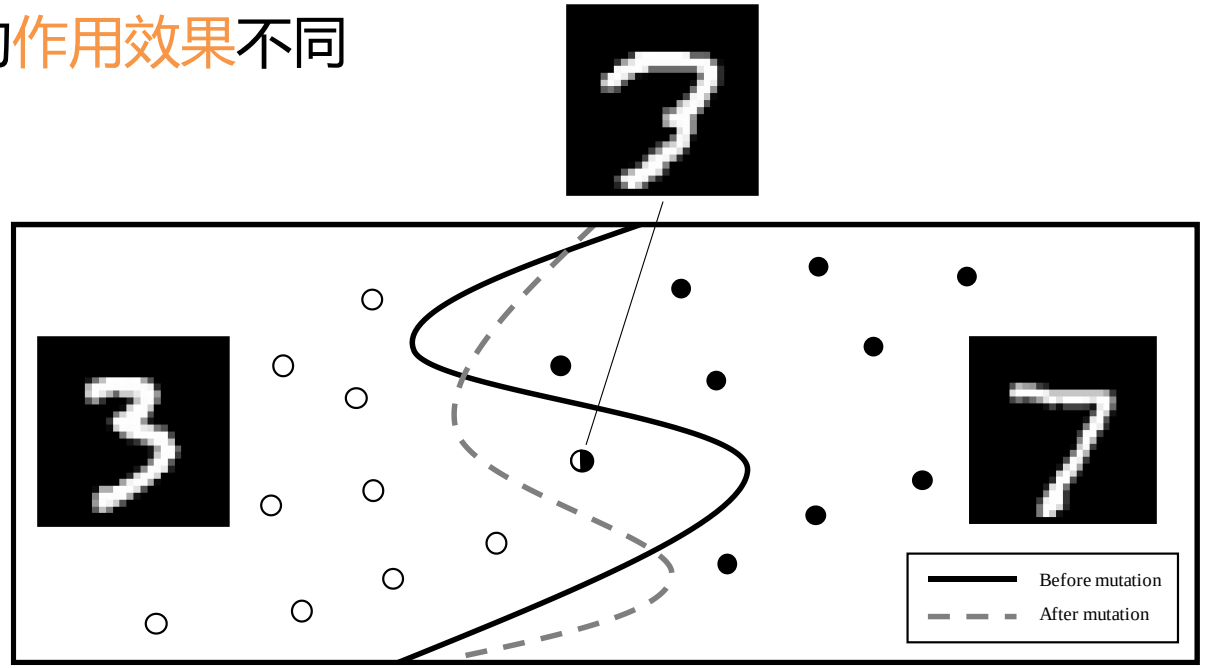
- 示例
  - 但是，对于Moutant 5、6 和 7：
    - **Moutant 5: if Age > 14 do: => if Age %%14 do:**
      - 语法错误突变
    - **Moutant 6:删除 Doctor = General Physician();**
      - 与输入无关突变
    - **Moutant 7:if Age > 14 do: =>if Age >14 & Age >14 do:**
      - 等效突变

Moutant 5	Moutant 6	Moutant 7
err	×	×
err	×	×
err	×	√

  - 如果在测试的最后阶段还有一些突变体存在，这意味着它可能是一个无效的突变也可能是数据集不充分。

历史发展

- **2018 年**
  - **Ma等人与Shen等人首次将突变测试引入智能化系统测试**
  - 核心思想
    - 位于分类边界附近的样本更可能引发模型错误
    - 不同的突变算子对分类边界的作用效果不同
  - 突变算子
    - 权重的高斯模糊
    - 神经元输出反转
    - 神经元输出值归零
    - 层删除
    - 层添加
    - ……



总结

总结

- 人工智能系统测试 - **Test For AI**
  - 测试属性 - **Why to test?**
  - 测试组件 - **What/Where to test?**
  - 测试流程 - **How to test?**

参考文献

- [1] Murphy C, Kaiser G E, Arias M. An approach to software testing of machine learning applic ations[J]. 2007.
- [2] Murphy C, Kaiser G E, Hu L. Properties of machine learning applications for use in metamo rphic testing[J]. 2008.
- [3] Pei K, Cao Y, Yang J, et al. Deepxplore: Automated whitebox testing of deep learning systems [C]//proceedings of the 26th Symposium on Operating Systems Principles. 2017: 1-18.
- [4] Ma L, Zhang F, Sun J, et al. Deepmutation: Mutation testing of deep learning systems[C]//20 18 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE). IEEE, 20 18: 100-111.
- [5] Shen W, Wan J, Chen Z. Munn: Mutation analysis of neural networks[C]//2018 IEEE Intern ational Conference on Software Quality, Reliability and Security Companion (QRS-C). IEEE, 20 18: 108-115.

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！