软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(5):1255−1275 [doi: 10.13328/j.cnki.jos.005951]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

深度神经网络测试研究综述*

王 赞[1], 闫 明[1], 刘 爽[1], 陈俊洁[1], 张栋迪[1], 吴 卓[2], 陈 翔[3]

[1](天津大学 智能与计算学部,天津 300350)

[2](天津大学 国际工程师学院,天津 300350)

[3](南通大学 信息科学技术学院,江苏 南通 226019)

通讯作者: 刘爽, E-mail: shuang.liu@tju.edu.cn

摘 要: 随着深度神经网络技术的快速发展、大数据的涌现和计算能力的显著提升,深度神经网络被越来越多地应用到各个安全攸关领域,例如自动驾驶、人脸识别、飞机碰撞检测等.传统的软件系统通常由开发人员手工编写代码实现其内部的决策逻辑,并依据相应的测试覆盖准则设计测试用例来测试系统代码.与传统的软件系统不同,深度学习定义了一种新的数据驱动的编程范式,开发人员仅编写代码来规定深度学习系统的网络结构,其内部逻辑则由训练过程获得的神经元连接权值所决定.因此,针对传统软件的测试方法及度量指标无法直接被移植到深度神经网络系统上.近年来,越来越多的研究致力于解决深度神经网络的测试问题,例如提出新的测试评估标准、测试用例生成方法等.调研了92篇相关领域的学术论文,从深度神经网络测试度量指标、测试输入生成、测试预言这3个角度对目前已有的研究成果进行了系统梳理.同时,分析了深度神经网络测试在图像处理、语音处理以及自然语言处理上的已有成果,并介绍了深度神经网络测试中应用到的数据集及工具.最后,对深度神经网络测试的未来工作进行了展望,以期为该领域的研究人员提供参考.

关键词: 深度神经网络;测试覆盖;测试用例生成

中图法分类号: TP311

中文引用格式: 王赞,闫明,刘爽,陈俊洁,张栋迪,吴卓,陈翔.深度神经网络测试研究综述.软件学报,2020,31(5):1255−1275. http://www.jos.org.cn/1000-9825/5951.htm

英文引用格式: Wang Z, Yan M, Liu S, Chen JJ, Zhang DD, Wu Z, Chen X. Survey on testing of deep neural networks. Ruan Jian Xue Bao/Journal of Software, 2020,31(5):1255−1275 (in Chinese). http://www.jos.org.cn/1000-9825/5951.htm

Survey on Testing of Deep Neural Networks

WANG Zan[1], YAN Ming[1], LIU Shuang[1], CHEN Jun-Jie[1], ZHANG Dong-Di[1], WU Zhuo[2], CHEN Xiang[3]

[1](College of Intelligence and Computing, Tianjin University, Tianjin 300350, China)

[2](International Engineering Institute, Tianjin University, Tianjin 300350, China)

[3](School of Information Science and Technology, Nantong University, Nantong 226019, China)

Abstract: With the rapid development of deep neural networks, the emerging of big data as well as the advancement of computational power, Deep Neural Network (DNN) has been widely applied in various safety-critical domains such as autonomous driving, automatic face recognition, and aircraft collision avoidance systems. Traditional software systems are implemented by developers with carefully designed programming logics and tested with test cases which are designed based on specific coverage criteria. Unlike traditional software

* 基金项目: 国家自然科学基金(61872263, 61802275, 71502125); 天津市智能制造专项资金(20191012); 天津大学自主创新基金(2019XZC-0073, 2020XZC-0042)

Foundation item: National Natural Science Foundation of China (61872263, 61802275, 71502125); Intelligent Manufacturing Special Fund of Tianjin (20191012); Innovation Research Project of Tianjin University (2019XZC-0073, 2020XZC-0042)

本文由“系统软件构造与验证技术”专题特约编辑赵永望副教授、刘杨教授、王戟教授推荐.

收稿时间: 2019-09-01; 修改时间: 2019-10-24; 采用时间: 2019-12-24; jos 在线出版时间: 2020-04-07

development, DNN defines a data-driven programming paradigm, i.e., developers only design the structure of networks and the inner logic is reflected by weights which are learned during training. Traditional software testing methods cannot be applied to DNN directly. Driven by the emerging demand, more and more research works have focused on testing of DNN, including proposing new testing evaluation criteria, generation of test cases, etc. This study provides a thorough survey on testing DNN, which summarizes 92 works from related fields. These works are systematically reviewed from three perspectives, i.e., DNN testing metrics, test input generation, and test oracle. Existing achievements are introduced in terms of image processing, speech processing, and natural language processing. The datasets and tools used in DNN testing are surveyed and finally the thoughts on potential future research directions are summarized on DNN testing, which, hopefully, will provide references for researchers interested in the related directions.

Key words: deep neural network; test coverage; test case generation

随着大数据和人工智能时代的到来,深度学习(deep learning,简称 DL)技术得以迅猛的发展,逐渐成为人工智能领域的关键技术.深度学习多采用由神经元互连构成的深度前馈神经网络,经过输入层、隐藏层、激活函数、输出层等,最终将神经元的输入映射到输出端.深度学习研究的浪潮始于 2006 年,Hinton 等人[1]发表了首篇关于深层网络的论文,标志着神经网络进入深度网络阶段.卷积神经网络(convolutional neural networks,简称 CNN)和循环神经网络(recurrent neural network,简称 RNN)是深度神经网络(deep neural networks,简称 DNN)的两种典型网络结构.CNN 使用卷积核提取图片特征,既解决学习的参数量过大的问题,又能很好挖掘局部特征,因此常被应用于图像处理应用.RNN 的特点则是能对时间序列上的变化进行建模,能够处理具有时序信息的数据,主要应用于语音处理、自然语言处理等领域.为了解决高识别率、实时性等具体需求,以 RNN 或 CNN 结构作为基础的、更为复杂的深度网络结构陆续被提出,例如长短期记忆网络(long short-term memory,简称 LSTM)、AlexNet 等.深度神经网络在解决人工智能中的一系列难题方面发挥了重要作用,在许多重要的应用中取得了显著的成功.在计算机视觉领域,深度学习技术极大地促进了图像分类、图像分割、目标识别以及图像风格迁移等方向的发展;在语音视频领域,深度学习对语音识别领域也产生了很大的影响,端对端的深度学习语音系统的应用,使得语音识别错误率取得大幅度的降低;在自然语言处理方面,深度学习技术已经被成功应用于文本挖掘、情感分析、机器翻译等方向;深度学习显著促进了强化学习领域地发展,基于深度学习技术的多智能体能够完美执行游戏对战等任务;深度学习也在其他科学领域表现出良好地应用性,包括基于深度学习的癌细胞识别技术、农业土壤监测技术、广告精准投放技术、金融欺诈识别、信用评价以及自动驾驶汽车等.

随着其广泛应用,深度神经网络系统的质量问题也被重点关注.由于深度神经网络的结构复杂,数据中微小的扰动,即便无法被人类发现,却可能造成深度神经网络做出错误的判断,进而输出错误的结果.更进一步,由于深度神经网络越来越多地被部署在自动驾驶汽车系统、恶意软件检测系统以及飞机碰撞避免系统等安全攸关领域,对这类 DNN 系统进行充分的测试并保证其质量至关重要.因此,迫切需要找到这些潜在的威胁来提高神经网络的安全性,使之能够应用于更多安全性要求高的场景.现阶段,部分研究人员通过对传统软件测试方法进行改进,生成适用于深度神经网络的测试用例,或者提出新的测试方法.同时,作为一种特殊的软件制品,深度学习框架的测试也得到了部分研究组的关注.

目前 Huang 等人[2]、Zhang 等人[3]和 Xiang 等人[4]对已有的机器学习和 DNN 相关的测试和验证方面的研究成果从不同方面进行了系统的总结.Huang 等人通过对近年来 DNN 测试相关论文整理和分析,重点从验证、测试、对抗攻击和防御以及 DNN 可解释性这 4 个方面对 DNN 的安全性和可信任性进行了细致地总结.Zhang 等人从更广的机器学习测试角度入手,对机器学习测试流程、测试组成、测试属性以及应用场景等方面对现有的研究成果进行梳理.Xiang 等人则对机器学习、自治系统和神经网络的形式化验证方法做出总结,并重点关注安全相关领域中系统的形式化验证技术.本文则主要关注深度神经网络测试相关的内容,充分调研深度神经网络测试的相关研究,关注在测试深度神经网络时,如何衡量网络的稳定性和测试充分程度,如何生成有效的数据用于测试,如何判断神经网络的输出是否符合预期,当前深度神经网络的测试已经在哪些领域被应用等方面,试图从测试度量指标、测试输入生成、测试预言、测试应用和评测数据集等方面系统梳理深度神经网络测试的相关工作,并对深度神经网络测试的未来给出展望,以此为该领域的研究人员提供一份参考.

本文所提及的深度神经网络研究框架如图 1 所示:测试度量指标包含测试覆盖指标和鲁棒性指标,测试覆

盖指标用于衡量测试输入对 DNN 系统的测试充分程度,而鲁棒性指标用于衡量 DNN 系统在不同输入下是否能稳定地做出符合预期的行为;测试输入生成是指生成用于测试 DNN 系统的输入的步骤,分为基于覆盖的输入生成和基于对抗性的输入生成;测试预言是指衡量 DNN 系统在各种输入下的表现是否符合预期的标准;测试应用是 DNN 测试在各领域的应用情况;评测数据集是对 DNN 系统测试常用的测试数据集的描述.

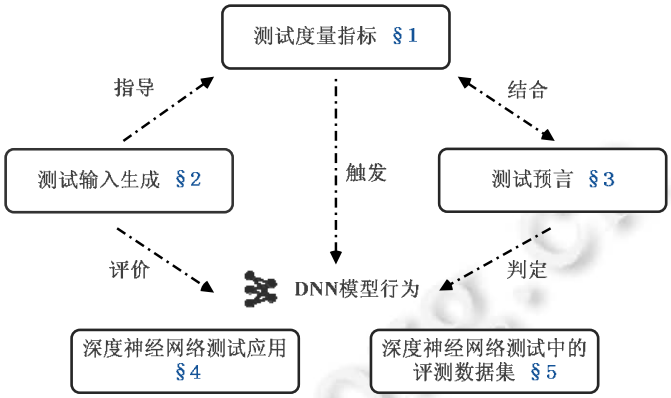

Fig.1 Research framework of deep neural network testing

图 1 深度神经网络测试的研究框架图

为了对该研究问题进行系统的梳理和分析,我们首先以“Deep Learning testing”,“Adversarial example”,“Artificial Intelligence testing”,“Deep Neural Networks testing”等设为主要搜索关键词,在国内外重要的学术搜索引擎(例如 Google 学术搜索、DBLP、CNKI 等)中检索出相关论文;随后,我们筛选并移除与该综述问题无关的论文;接着,通过查阅论文中的相关工作和研究人员的已发表的论文列表,以及通过已搜索到的文献的引用和被引用获取更多相关文献;最终,我们确定了与该综述研究问题直接相关的论文 92 篇、相关评测数据集及工具 28 个.论文在不同年份的发表数分布如图 2 所示,其中选中的部分论文发表在 CCF 评级为 A,B 的各领域会议或期刊上,影响力较大,具有一定的权威性.收录的论文在 CCF 评级的分布如图 3 所示,其中:软件工程及系统和程序语言领域(20 篇)、网络与信息安全领域(10 篇)、计算机科学理论领域(2 篇)、人工智能领域(10 篇).除此之外,我们收录的文献来源还包括 arxiv(17 篇)、CCF 中的 C 类会议期刊以及 CCF 未收录的会议和期刊(33 篇).图 4 展示了综述不同章节(如图 1 所示)中的论文分布情况.

本文第 1 节对 DNN 测量度量指标进行总结.第 2 节对测试输入生成已有的方法进行总结梳理,从基于覆盖的测试用例生成技术和基于对抗性的测试输入生成两个方向进行分析.第 3 节对测试预言进行梳理,主要从蜕变测试和差异测试入手进行分析.第 4 节是对 DNN 测试应用的总结,分为 4 个领域的应用:图像处理领域、语音识别领域、自然语言处理领域和其他领域.第 5 节是对文献中所使用到的评测数据集进行的总结.最后总结全文,并展望未来可能的研究方向.

Fig.2 Quantity distribution of articles in different years

图 2 论文在不同年份的文献数量分布

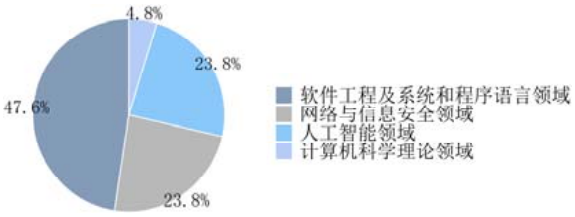

Fig.3 Research field distribution of articles (CCF A and B)

图 3 论文收录 CCF A 类、B 类会议期刊的分布情况

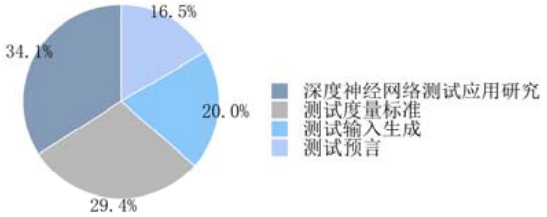

Fig.4 Proportion of articles in key chapters

图 4 论文重点章节中的论文分布比例

1 DNN 测试度量指标

测试度量指标主要用于评估测试用例集的充分性,基于传统软件的测试发展多年,形成了一套相对成熟的测试方法以及测试度量体系(例如语句覆盖、判断覆盖、条件覆盖等).但是鉴于深度神经网络的特性,针对传统软件的测试度量并不能完美地移植到深度学习测试领域.深度学习定义了一种全新的数据驱动的编程范式,开发人员编写 DNN 模型的训练程序,而后通过对一组训练数据的学习来构造 DNN 系统的内部逻辑.现有的深度学习度量指标主要分两类:DNN 模型测试覆盖度量指标和鲁棒性度量指标.顾名思义,测试覆盖指标通过计算测试用例的覆盖率来评估测试输入对于神经网络测试的充分性,而鲁棒性度量指标则重点关注衡量 DNN 模型在各种输入数据下表现的稳定性.除此之外,还有一些针对 DNN 系统的安全性等其他方面度量的研究.

1.1 DNN测试覆盖指标

部分学者认为,研究 DNN 系统内神经元的激活值分布对于发现 DNN 系统的边界行为具有重要作用.受传统测试覆盖思想的启发,他们通过统计和追踪神经元激活值的分布、或相邻层神经元之间激活值的变化关系,提出了基于神经元激活值的结构化测试覆盖指标,以此定义测试输入对于 DNN 系统的覆盖率,并提出了基于覆盖引导的测试输入生成方法.除此之外,也有研究人员提出了通过计算测试用例相对于训练数据的差异程度来探究测试用例充分性相关的工作.表 1 罗列了现有测试覆盖准则的简要信息.

Table 1 Summary of existing coverage criteria

表 1 现有覆盖度量总结

文献来源	发表年份	覆盖度量	指导测试用例生成
DeepXplore[5]	2017	神经元覆盖	是
DeepCover[6]	2018	符号-符号覆盖、距离-符号覆盖、符号-值覆盖、距离-值覆盖	是
DeepGauge[7]	2018	*k*-多区域神经元覆盖、神经元边界覆盖、强神经元激活覆盖、Top-*k* 神经元覆盖、Top-*k* 神经元模式	是
DeepCT[8]	2018	*t*-way 组合稀疏覆盖、*t*-way 组合密集覆盖、(*p*,*t*)-完整性覆盖	是
DeepPath[9]	2019	*l*-SAP 覆盖、*l*-OAP 覆盖、*l*-FSP 覆盖	否
DeepInspect[10]	2019	神经元激活概率向量距离、平均偏差等度量	否
DeepCruiser[11]	2018	状态级别覆盖、转换级别覆盖	是
SADL[12]	2019	意外覆盖	否

Pei 等人提出了首个针对真实世界 DNN 系统的白盒测试框架 DeepXplore[5],并率先引入了神经元覆盖(neuron coverage)作为度量指标.Pei 等人认为,现有的 DNN 测试技术存在两种局限性:1) 采用随机测试,过分依赖手工标记的测试数据,同时,某些领域的测试输入不易标记和扩展;2) 由于缺少针对 DNN 系统的测试方法与测试覆盖标准,测试集的覆盖率较低,难以检测错误的边界输入.为了解决上述问题,Pei 等人设计了神经元覆盖率指标,并提出了针对 DNN 系统的差异测试方法,该方法寻找使一组功能相同 DNN 模型产生差异行为且达到高神经元覆盖率的测试输入.该过程被建模为联合优化问题,并采用基于梯度下降的搜索技术求解.Pei 等人在 MNIST[13],ImageNet[14],Driving[15]等 5 个数据集上对 15 种 DNN 模型进行实验.实验表明:使用 DeepXplore 生成

的测试输入重新训练 DNN 系统,可以将模型准确率提高 3%.

Sun 等人在文献[6]指出:神经元覆盖粒度较粗,一些简单测试套件可以轻易达到 100%的覆盖率.为了解决上述问题,Sun 等人借鉴传统软件测试领域 MC/DC 标准的思想,根据相邻层之间神经元激活值的变化情况,提出了符号-符号覆盖、距离-符号覆盖、符号-值覆盖以及距离-值覆盖这 4 种 DNN 测试度量以及相应的基于线性规划的测试用例生成方法,并在工具 DeepCover 中实现了上述算法.Sun 等人在 MNIST 数据集上针对 10 个 DNN 系统展开实验,并从缺陷发现、DNN 安全性分析、测试充分性和 DNN 中间结构分析这 4 个方面证明其提出的测试覆盖标准与测试用例生成方法的有效性.

在 Pei 等人研究的基础上,Ma 等人进一步扩展了神经元覆盖的概念,并首次提出了一组针对 DNN 系统的多粒度测试标准 DeepGauge[7].他们认为:DeepXplore 中提出的神经元覆盖粒度较粗,对所有的神经元采取了相同的激活阈值,没有考虑到不同神经元的输出统计分布之间的差异性.DeepGauge 采用统计不同神经元的输出并动态地分配神经元激活阈值的方法定义多粒度的神经元覆盖率,分别从神经元级别和层级别两个层次入手,引入包括 *k*-多区域神经元覆盖、神经元边界覆盖和 Top-*k* 神经元覆盖在内的 5 种覆盖标准.Ma 等人分别在 MNIST 与 ImageNet 上,采用 FGSM[16],BIM[17],JSMA[18]和 C&W[19]这 4 种对抗性样本生成方法与 5 种 DNN 模型进行实验,并与 DeepXplore 进行对比.在 DeepXplore 中,部分原始数据和简单的对抗性样本即可实现接近 100%的覆盖率.而在 DeepGauge 提出的覆盖指标上,所达到的覆盖率有明显差异.相比之下,DeepGauge 捕捉原始测试数据与对抗性测试数据差异的能力更强.

Ma 等人受到传统组合测试方法的启发,首次提出了针对 DNN 系统的组合测试方法 DeepCT[8],并提出一系列测试覆盖准则及相应测试用例生成方法.他们将神经网络中神经元的输出值进行离散化,基于组合测试的思想定义了两种 *t*-way 组合测试覆盖标准,并提供了基于约束求解的测试用例生成方法.该方法使用两个 DNN 模型在 MNIST 数据集上,与随机测试方法进行对比实验.结果表明:相比于随机测试方法,DeepCT 在两个测试覆盖标准上均获得更高的覆盖率,同时检测到更多的对抗性样本.但由于真实的 DNN 系统具有相当多的神经元,即便使用 CPLEX[20]这样的的高性能约束求解器,基于约束求解的 DeepCT 在对真实的 DNN 系统进行测试时仍需要很高的代价.

Sekhon 等人[21]提出:深度神经网络缺乏明确的控制流结构,使其无法应用代码覆盖率等传统的软件测试度量指标.因此,他们借鉴了 MC/DC 以及组合测试的思想,综合考虑相邻层神经元以及特定层神经元激活情况,提出了针对 DNN 模型的 2-way 覆盖准则和基于该覆盖标准的测试输入生成方法.他们使用 LeNet-1,LeNet-4 以及 LeNet-5 模型在 MNIST 数据集上进行实验.实验结果表明:Sekhon 等人提出的覆盖度量在规模较小(例如 10 张图片)的数据集上覆盖率较低,并能够随着数据集规模增大显著提升;同时,其对应的测试输入生成方法能够很好地产生对抗性样本,并在随着测试集中对抗性样本的加入而达到较高的覆盖率.但是,该覆盖度量对来自真实世界的更大规模 DNN 系统的可扩展性及其对不同神经网络架构的适应性还有待测试.

受到面向路径的测试方法的启发,Wang 等人[9]提出了一组针对 DNN 模型的路径驱动的测试度量指标 DeepPath.DeepPath 借鉴传统软件工程中路径的概念,将模型中的单一神经元视为节点,将不同层之间的神经元连接视为路径,对 DNN 中的子路径进行了定义,并提出了 3 种路径覆盖度量,*l*-SAP,*l*-OAP 和 *l*-FSP.他们使用 LeNet-3,LeNet-5 以及 LeNet-10 这 3 种模型,在 MNIST 数据及上展开实验,并与 DeepXplore 展开对比.Wang 等人通过实验表明:改变激活阈值,DeepPath 能够在覆盖率上得到更大程度的提升.同时,随着输入中对抗性样本的加入,DeepPath 产生的覆盖率有显著的改变,从而表明其能够更好地识别对抗性样本.

Tian 等人[10]认为,现有的 DNN 测试技术无法很好地检测图片分类器在分类结果中的混淆和偏差.因此,他们提出了白盒测试框架 DeepInspect,从而自动检测基于 DNN 的图片分类器的混淆和偏差错误.DeepInspect 依据模型中的激活神经元的路径,定义了 NAPVD(neuron activation probability vector distance)、平均偏差(average bias)等度量指标.Tian 等人分别针对单一标签和多标签分类任务,使用 ResNet-18,ResNet-50 等图片分类模型,在 MNIST,CIFAR-10/CIFAR-100[22],COCO[23]等数据集上展开实验.

实验分析结果表明,现有的 DeepXplore 以及 DeepGauge 中的覆盖率指标,针对不同类别的输入得到的覆盖

率是相似的,并不能有效地区分不同类别的输入,从而无法帮助识别模型的混淆和偏差错误.DeepInspect 中的指标则可以区分不同类别的输入并帮助识别 DNN 模型的混淆以及偏差错误.

上述测试覆盖指标主要依据 CNN 网络结构提出.由于 RNN 中存在回路以及网络内部状态之间的转移,先前的测试覆盖指标针对 RNN 网络不具有较好的可移植性.因此,Du 等人提出了状态级别和转换级别两种基于抽象状态转换模型的 RNN 网络测试覆盖准则,以捕获其动态的状态转换行为,并基于此标准提出了一个能够系统生成大规模测试输入的自动化测试框架 DeepCruiser[11],通过覆盖引导生成测试用例来揭示有状态的 DNN 系统的缺陷.Du 等人在 DeepSpeech-0.3.0[24]预训练模型上展开实验,使用 Fisher[25],LibriSpeech[26],Switchboard[27]以及英语普通版语音训练语料库[28]进行训练.实验表明:DeepCruiser 能够根据覆盖范围反馈生成具有高覆盖率的测试用例,并有效地对基于 RNN 的自动语音识别系统进行缺陷检测.

除此之外,Kim 等人展开了关于深度神经网络测试充分性方面的工作[12].他们认为,测试充分性准则应该同样适用于单个的测试输入,并在论文中指出:现有的覆盖标准都不是细粒度的,对于单个测试输入信息的分析较少.为了解决上述问题,Kim 等人针对 DNN 系统提出了一种细粒度的测试充分性框架 SADL,用于计算测试输入相对训练输入的“意外值”,并引入意外充分性(surprise adequacy,简称 SA)以及意外覆盖(surprise coverage,简称 SC)两种指标.SADL 提供了基于核密度估计(kernel density estimation,简称 KDE)和基于欧氏距离的两种计算 *SA* 的方法,通过比对测试输入与训练数据集的 *SA* 值差异来表示测试输入相对训练数据的意外值.实验结果表明:*SA* 和 *SC* 能够准确地捕捉输入的意外值,对于区分对抗性样本和指导模型重训练有重要作用.

针对将传统的程序覆盖理论应用到深度学习模型上的想法,Li 等人[29]提出:由于神经网络与传统编程方式的不同,现有的测试覆盖准则可能具有误导性.他们认为,对抗性输入和正确输入在输入空间中的共存打破了传统结构化覆盖标准中的同质性,并认为面向覆盖的搜索方法和面向对抗的搜索具有相似性.他们认为:与面向对抗的研究相比,现有的度量标准并没有提供更多的关于模型质量的信息.Li 等人通过 3 组实验验证他们的猜想并初步得到如下结论:之前提出的神经网络结构覆盖标准对于具有真实输入的 DNN 系统的故障检测可能是无效的.他们认为,一个有效的度量准则应该能在具体的使用场景下具有以下特点:1) 在自然应用场景下,具有区分相同规模和不同错误率的自然样本集的能力;2) 在对抗性样本生成场景下,相比面向对抗的搜索策略,具有更高效地生成对抗性样本的能力.

1.2 DNN鲁棒性度量指标

DNN 系统的鲁棒性体现了其在各种正常和异常输入下的稳定程度.在本文中,DNN 鲁棒性度量指标是对 DNN 表现稳定程度的一类度量.而一些相关工作[30,31]从安全性角度对 DNN 的表现稳定程度进行更细的分类:对于输入 x 和包含 x 的输入子空间 X,如果 X 中的任意输入的在 DNN 下的分类结果都与 x 相同,则认为 DNN 具有安全性.安全性用于判断 DNN 在一个输入子空间下的表现是否稳定,在本文的分类中,安全性亦是用于衡量 DNN 的稳定程度,因此被统一分类到 DNN 鲁棒性度量指标的类别中.

Lipschitz 连续性(Lipschitz continuity)[32]常被用于衡量 DNN 的鲁棒性,Lipschitz 连续性的定义如下:对于神经网络 N,$v[x]$为网络在输入 x 下的输出,如果存在常数 $c\geqslant 0$,满足在任意输入 x_1,x_2 下,$\|v_1[x]-v_2[x]\|\leqslant c\cdot\|x_1-x_2\|$,则称神经网络 N 具有 Lipschitz 连续性,其中,c 称为 Lipschitz 常数.

Lipschitz 连续性被用来定义深度神经网络的全局鲁棒性[33],直观解释为:当输入的差异被控制在一个较小范围之内($\|x_1-x_2\|$),神经网络的输出差异($\|v_1[x]-v_2[x]\|$)也控制在(由 Lipschitz 常数控制的)一定范围之内.

Xu 等人[34]提出计算 Lipschitz 常数的上界,以估计 DNN 的鲁棒性.但该上界过于宽泛,无法很好地反映 DNN 的鲁棒性.为了对 DNN 鲁棒性做出更加准确的衡量,Weng 等人[35]基于极值理论,提出了一种 DNN 鲁棒性指标 CLEVER.虽然 CLEVER 能很好地反映 DNN 的鲁棒性,但其需要针对每一个输入进行计算,因此计算过程过于复杂,难以进行实际的应用.Katz 等人[36]提出了对抗鲁棒性的概念,即模型能够正确分类通过微小扰动生成的攻击样本的能力.众多研究者基于对抗鲁棒性的定义,设计了一系列计算复杂度低且更能反映 DNN 在实际情形下的鲁棒性度量指标.Papernot 等人[37]采用了一种与 Lipschitz 连续性相似的概念,更好地实现对 DNN 鲁棒性的定量度量.该度量可以描述为输入样本与其在输入空间中最近的对抗样本的距离,其定义为

$$\rho_{adv}(F)\simeq\frac{1}{|\chi|}\sum_{X\in\chi}\min_{\delta X}\|\delta X\|,$$

其中,χ为样本空间,δX 是对原输入样本的扰动.由于寻找样本空间中距离最近的对抗样本需要尝试所有的对抗样本,这在实际情况下并不可行.因此该方法在实际操作中,在原样本基础上生成 9 个对抗样本,并找到距离最近的对抗样本作为 $\min_{\delta X}\|\delta X\|$.该方法大幅度降低了神经网络鲁棒性衡量方法的计算复杂度,得到的结果能够在一定程度上反映神经网络在对抗攻击下的稳定程度.但是,上述度量过程中的 9 个对抗样本是基于某个特定对抗样本生成方法生成的,这意味着此度量标准只能描述网络在特定种类对抗攻击下的鲁棒性.Mangal 等人认为:以 Lipschitz 连续性为代表的现有的 DNN 鲁棒性指标考虑的是输入空间中的最差情况,这对于衡量 DNN 在实际情形下的鲁棒性过于严格,并且类似指标的计算过程代价过高.参考 Lipschitz 连续性,Mangal 等人设计了一种基于概率的鲁棒性指标[38],称为概率鲁棒性,其定义为:对于输入分布 D 中的输入 x,x',神经网络 v 的输出应该满足 $\Pr_{x,x'\sim D}(\|v[x]-v[x']\|\leqslant k*\|x-x'\|\,|\,\|x-x'\|\leqslant\delta)\geqslant 1-\varepsilon$,即当输入的差异被控制在较小范围内时,神经网络对应的输出被控制在一定范围内的概率应该大于等于特定阈值 $1-\varepsilon$.概率鲁棒性在实际的非对抗情形下是一种更加高效合理且计算代价更小的指标.部分研究者使用局部 Lipschitz 连续性代替全局 Lipschitz 连续性作为 DNN 的鲁棒性指标,即分析某个输入样本下的 Lipschitz 连续性代替整个输入空间下的 Lipschitz 连续性,以降低计算代价,并获得更加准确的鲁棒性边界.Weng 等人[35]通过对某个输入样本的周围空间进行采样,然后基于局部 Lipschitz 连续性对鲁棒性进行估计.由于采样的数量影响该估计的准确性,他们通过极值理论证明了估计值服从极值分布,保证了有限采样下的估计准确性.

除了 Lipschitz 连续性,一些工作从 DNN 的另一些特征出发,提出了多种度量 DNN 鲁棒性的指标.Bastani 等人[39]提出了基于对抗性样本出现的频率和对抗性样本的严重的 DNN 鲁棒性度量指标,并设计了一种新的基于鲁棒性编码的 DNN 鲁棒性度量算法 A_{LP} 算法.他们在 LeNet 和 NiN 两个网络上展开实验,分别使用 A_{LP} 和 $A_{L\text{-}BFGS}$(采用 L-BFGS 方法生成对抗性样本,并以此度量神经网络鲁棒性)指导对抗性样本产生,并利用对抗性样本对模型进行重训练,以提升模型鲁棒性.实验结果证明:相比于已有算法,A_{LP} 算法对于模型鲁棒性的评估更加地精确,A_{LP} 算法能有效避免模型对于算法产生的对抗性样本过拟合的问题.Cheng 等人[40]针对 DNN 系统,从鲁棒性、可解释性、完整性和正确性这 4 个方面提出了一套度量指标 RICC.他们指出:如果将传统软件测试领域度量指标,例如 MC/DC 等,引入到 DNN 中,可能会导致测试分支数目随着模型规模的增加呈指数爆炸增长.同时,他们认为:单一神经元的激活值,如 Pei 等人在 DeepXplore 中所提出的神经元覆盖度量,与整个网络输出结果的联系性不强.基于上述原因,Cheng 等人从鲁棒性、可解释性、完整性和正确性这 4 个方面提出了 8 种度量指标.但是,Cheng 等人没有在主流数据集上进行实验来阐述他们的度量在测试用例生成以及对抗性样本检测方面的作用,同时也没有提出相应的测试用例生成方法.

2 测试输入生成

与测试传统软件类似,在对 DNN 系统的测试中,使用足够的测试输入对系统的一般行为和各种边界条件下的行为进行充分的测试是必要的.我们将当前 DNN 系统测试输入生成的研究工作分为两类:第 1 类方法通常从软件工程的角度出发,将传统软件测试的思路迁移到 DNN 模型的测试中,通过对给定的种子输入进行指定的变换,以最大化模型覆盖率为目标来生成测试输入,我们称这类方法为基于覆盖的测试输入生成方法;第 2 类则从机器学习和深度学习的角度入手,通过向原始样本添加微小扰动的方式产生对抗样本,使 DNN 系统进行错误分类,我们称此类方法为基于对抗的测试输入生成方法,包括白盒方法与黑盒方法.基于覆盖的方法更关注生成的测试输入对 DNN 内部状态的影响,即测试输入是否对网络内部状态实现了测试覆盖.基于对抗的方法则更关注生成的测试输入是否能够使 DNN 产生错误输出.图 5 展示了不同输入生成测试方法类别间的关系.表 2 总结了各种类别的输入生成方法的评价方法、指导目标和实验数据集.

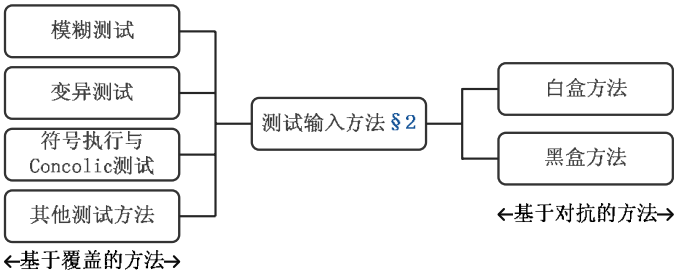

Fig.5 Test input generation method

图 5 输入测试生成方法

Table 2 Summary of test input generation methods

表 2 测试输入生成方法总结

方法	类别	评价方法	指导目标	实验数据集
DLFuzz[41]	模糊测试	神经元覆盖率、生成效率、图片质量	神经元覆盖	MNIST、ImageNet
TensorFuzz[42]	模糊测试	数值误差、预测精度	激活向量	MNIST
DeepHunter[43]	模糊测试	测试精度、测试覆盖率	神经元覆盖、*k*-多区域神经元覆盖、神经元边界覆盖、强神经元激活覆盖、top-*k* 神经元覆盖	MNIST、CIFAR-10、ImageNet
DeepConcolic[44]	Concolic 测试	测试覆盖率、对抗性样本质量	符号-符号覆盖、神经元边界覆盖、神经元覆盖、Lipschitz 连续性	MNIST、CIFAR-10
DeepCheck[45]	符号执行	–	像素重要性、像素变化个数	MNIST
DeepCT[8]	组合测试	组合测试覆盖率、模型准确率、对抗样本数量	*t*-way 组合稀疏覆盖、*t*-way 组合密集覆盖、(*p*,*t*)-完整性覆盖	MNIST
FGSM[16]	对抗样本生成	分类准确率	像素值差异	MNIST
Xiao 等人[46]	对抗样本生成	人类对比评价	像素平移距离	MNIST、CIFAR-10、ImageNet
OptMargin[47]	对抗样本生成	分类准确率、失真度	像素值差异	MNIST、CIFAR-10
边界攻击[48]	对抗样本生成	分类准确率、与原样本距离	像素值差异	MNIST、CIFAR-10、ImageNet
Zhao 等人[49]	对抗样本生成	人类对比评价	样本在输入空间的分布	MNIST、LSUN[50]
Wicker 等人[51]	对抗样本生成	分类置信度	蒙特卡罗树搜索	MNIST、CIFAR-10、Nexar、ImageNet

2.1 基于覆盖的测试输入生成

我们将已有的基于覆盖的测试输入方法分为模糊测试、变异测试、符号执行与 Concolic 测试以及其他测试方法.

2.1.1 模糊测试

模糊测试[52]是一种常用的高效的软件测试方法,在各个测试领域被广泛采用.模糊测试通过将种子输入随机或者按照某种规则进行变换作为新的输入,并观察软件在这些非预期输入下是否会发生错误.近些年来,一些研究者将模糊测试应用到 DNN 系统的测试用例生成中并取得了不错的效果.

Guo 等人[41]提出了第一个差异模糊测试框架 DLFuzz,用于指导 DNN 系统暴露异常行为.该工作旨在最大化神经元覆盖率,并为给定的 DL 模型生成更多的对抗性输入,而无需参照其他 DL 模型或手动标记.DLFuzz 迭代选择有价值的神经元来覆盖更多的逻辑,并通过应用微小的扰动来改变测试输入.在突变过程中,有助于在后续的模糊过程中增加神经元的覆盖范围的突变输入将会被保留.为了最大化神经元覆盖,他们还提出了 4 种启发式策略来选择更有可能提高覆盖率的神经元.该方法通过数据集 MNIST 和 ImageNet 上训练的 6 个 DNN 系统与 DeepXplore 进行了对比.从效率上看,在相同时间内,DLFuzz 平均神经元覆盖率比 DeepXplore 高 1.10%至 5.59%,平均时间内 DLFuzz 比 DeepXplore 多产生 338.59%的对抗性输入.从图片质量上看,DeepXplore 生成的扰

动是容易察觉的,而 DLFuzz 生成的扰动是难以察觉的.另外,DLFuzz 不用额外寻参考具有相似功能的其他 DL 模型,也不需要人工进行标记,因此能够节省 20.11%的时间消耗.

Odena 等人[42]将软件工程中的覆盖引导模糊测试概念引入神经网络领域,提出了 TensorFuzz,并基于实证研究验证了 TensorFuzz 的实用性.他们使用模型中神经元的激活值构成的激活向量作为覆盖指标,并使用近似最近邻算法判断激活向量是否已被覆盖,以此指导测试用例生成.Odena 在实证研究中发现:与随机搜索方法相比,TensorFuzz 产生的输入能够发现缺陷模型中的数值型错误,检测原始模型及其量化模型之间的不一致性,快速找到数值误差;除此之外,TensorFuzz 还具有发现 RNN 模型异常行为的能力.

Xie 等人[43]提出了一个基于覆盖引导的自动化模糊测试框架 DeepHunter,用于检测 DNN 模型的潜在缺陷.传统的基于覆盖引导的模糊测试,对系统的输入进行随机突变来检测软件中的边界情况,仅仅能检测到浅层的错误,例如格式解析错误等.然而,大多数 DNN 系统要求输入是特殊格式,传统的覆盖引导模糊测试方法并不适用.同时,针对 DNN 模型的测试,旨在检测模型功能而不是对 DNN 系统的代码进行漏洞检测.因此,Xie 等人使用 6 种测试覆盖标准引导 DNN 模糊测试,实现了 DNN 模型在开发和部署两个阶段的缺陷检测.DeepHunter 从初始种子输入中选取特定输入,进行数据变异生成新的测试用例,记录每一种覆盖标准下变异测试用例对应的覆盖率作为反馈,指导后续的测试用例生成并不断迭代.DeepHunter 在 MNIST,CIFAR-10,ImageNet 这 3 个数据上,使用 7 个 DNN 模型展开实验,阐述了 DeepHunter 对于 DNN 模型评估、DNN 系统错误行为检测以及 DNN 模型部署缺陷检测等方面有效性.

2.1.2 变异测试

变异测试[53]是一种评估用于测试用例集质量的重要手段.传统软件中的变异测试对代码进行变异,形成大量存在潜在问题的代码,称为变异体,产生这些变异体的不同变异规则与操作称为变异算子.测试用例集能够将多少变异体的错误暴露出来,可以作为测试用例集质量的衡量.严格来说,变异测试旨在评价测试数据的质量,并没有生成新的测试输入,但是对测试用例集质量的评价,对进一步地生成更加有效的测试输入具有重要的指导意义,因此我们将针对 DNN 系统的变异测试归类为测试输入生成.在对 DNN 系统的测试中,如何衡量测试的充分性,以及如何生成更加高质量、更容易暴露错误的测试数据是一个研究重点.将变异测试应用到对人工智能系统的测试中,可以帮助衡量已有数据集对人工智能系统的测试覆盖程度.目前,已有研究在这个方面取得了进展.

Ma 等人[54]借鉴传统软件变异测试的思想,将其应用到针对 DNN 系统的测试中,提出了 DeepMutation.DeepMutation 从源码级别和模型级别两个方面入手,分别提出了 8 种变异算子,对 DNN 系统的训练程序、训练数据以及模型文件进行变异,并提出了新的变异测试度量.Ma 等人在 MNIST 和 CIFAR-10 数据集上,使用 3 种 DNN 模型展开实验.实验结果表明:DeepMutation 可以对测试用例集质量进行有效的定量分析,帮助开发人员提高测试数据能力.对于变异测试,变异算子的变异有效性是测试效果的重要保证.Ma 等人在论文中提出:下一步工作将聚焦于如何提出更加有效的变异算子,使得变异算子能在程序中引入类似于人为错误的故障.

2.1.3 符号执行与 Concolic 测试

符号执行是软件测试中为了达成更高代码覆盖率而采用的一种测试输入生成方法.符号执行技术通过分析程序,计算出能够执行代码特定部分的程序输入,因此,通过符号执行技术产生的程序输入能够更加高效地实现代码覆盖.Concolic 测试技术[55]是一种将程序具体执行与符号执行结合起来的软件测试技术.直接执行程序能够以更小的代价实现对特定输入的测试,而将符号执行作为具体执行过程中的指导能够帮助以更少的执行次数发现错误,二者的结合能够发挥各自优势,以更高效率生成高质量的测试输入.

部分研究者将符号执行技术应用到了 DNN 系统的测试中.通过符号执行技术分析 DNN,计算使特定神经元产生特定输出的输入条件,从而生成能够满足更高覆盖率的测试输入.也有部分研究者将 Concolic 技术应用 DNN 系统的测试输入生成方面,以减少单纯依靠符号执行带来的测试输入生成的开销.

Sun 等人[44]将 Concolic 测试应用到 DNN 测试,提出了 DeepConcolic.他们在论文中使用 DeepXplore,DeepCover 等诸多工作中提出的 DNN 测试覆盖标准,将基于启发式逻辑的具体执行和符号执行相结合,迭代地

产生具有更高覆盖率的测试输入.DeepConcolic 在 MNIST 与 CIFAR-10 数据集上与 DeepXplore 进行了对比,结果表明,DeepConcolic 在达到更高的覆盖率的同时具有更快的速度,以此证明了 Concolic 测试在深度神经网络测试上的有效性.

Gopinath 等人[45]认为:现有的 Concolic 测试工作主要通过定义并实现相应的测试覆盖率来指导产生测试输入,而没有关注输入图片中那些易被攻击的重要像素,没有使用形式化方法对图片中的重要像素进行识别.因此,他们在 DeepCheck 介绍了一种 DNN 轻量级符号执行的新技术,并将其应用于图像分类算法的测试,以解决 DNN 分析中的两个具有挑战性的问题——重要像素的识别以及创建 1 像素和 2 像素攻击.DeepCheck 将给定的 DNN 转换为语义等效的命令程序,使用 3 个指标(*abs*,*co*,*coi*)量化像素对于图像分类标签的重要性,从而识别图像的重要像素.通过将问题建模为约束求解问题,对 DNN 模型进行 1 像素或 2 像素攻击,以产生与原图像有相同的激活模式、但分类结果不同的测试输入.他们在 MNIST 数据集上展开了实验,实验结果证明:DeepCheck 对于数据集中的每张图片均可产生 1 像素和 2 像素攻击样本;同时,DeepCheck 具有识别图像中重要可攻击像素的能力.

2.1.4 其他测试方法

组合测试是以系统中各个参数的取值组合作为覆盖标准的测试方法.组合测试也可以被应用到对深度神经网络的测试中,帮助生成测试输入.Ma 等人受到传统软件组合测试方法启发,首次提出了针对 DNN 系统的组合测试方法 DeepCT,并提出一系列用于组合测试方法的测试覆盖准则及相应测试用例生成方法[8].通过离散化神经网络中神经元的输出值,Ma 借鉴组合测试的思想定义的两种 *t*-way 组合测试覆盖标准并以此为指导,使用约束求解生成神经网络的测试用例.Li 等人认为:DNN 系统的测试不能脱离其应用所处的上下文,应该在真实的测试输入空间中进行操作性测试.在之前研究[29]的基础上,Li 等人[56]提出了有效的 DNN 测试方法.该方法利用 DNN 网络学出的分布表示,约减测试输入空间,有效缩减测试数据的标注代价.Li 等人通过实证研究验证了:相比原始随机选择的测试集,该方法能够大幅减少测试样本数量的情况下得到相同的网络测试精度.

2.2 基于对抗性的测试输入生成

另外一部分针对深度神经网络的测试输入生成的研究基于 2013 年 Szegedy 等人[33]提出的对抗样本这一重要概念.在图像分类领域,对图片输入 x 添加微小的扰动形成新的图片 $x+\Delta x$(其中,$||\Delta x||<\varepsilon$,ε是一个足够小的值)应该与原图片属于同一个分类.但是 Szegedy 等人发现:对于深度神经网络实现的图片分类器,这一假设并不成立.一些非随机的方式(比如通过优化的方法)可以比较容易地生成与原图像 x 分类结果不同的样本 $x+\Delta x$,这类样本被称为对抗样本.对抗样本对于 DNN 系统的测试具有重要作用,DNN 系统在对抗样本下的精确度可以作为 DNN 系统稳定性的衡量指标,使用对抗样本对 DNN 网络进行训练有助于提高 DNN 网络的性能和稳定性.针对深度神经网络的对抗样本研究需要两方面的内容——对抗样本需要与原样本具有更小的差异,因此难以人工或自动地与正常输入区分开;同时,对抗样本的生成方法需要对各种类型的深度网络都具有较高的成功率.

对抗样本相关的研究是当前的研究热点之一,最近几年有大量相关研究成果产生.一部分工作在生成对抗样本过程中需要获得 DNN 网络的内部状态,我们将它们分类为白盒方法;另一部分工作在生成对抗样本过程中只需要获得 DNN 网络在各种输入下对应的输出,而不需要关注其内部状态,我们将它们分类为黑盒方法.

2.2.1 白盒方法

白盒方法是指通过获得深度神经网络的内部状态,帮助生成对抗样本的一类方法,这类方法利用了深度神经网络的内部信息,通常更容易生成高质量的对抗样本.

部分现有的对抗样本生成工作利用神经网络内部状态构建优化函数,基于梯度下降等方式对函数进行优化,我们将这类工作称为基于优化的方法.基于优化的方法是对抗样本生成研究中最常用的方法.监督学习中,神经网络的训练过程可以看作对某个损失函数进行优化,使之结果最小的过程.与之对应的,基于优化的对抗样本生成方法将神经网络的预测值与原始标签之间的差异定义为目标函数,并对该函数进行优化,以得到具有轻微扰动,同时使神经网络进行错误分类的对抗样本.最早发现对抗样本的 Szegedy 等人给出了一个对抗样本生成方法,称为有边界约束的 L-BFGS[33]:假设 $f(x)$代表分类器 f 对图片 x 分类的结果,通过最小化 $c|\Delta x|+loss_f(x+\Delta x,l)$

的值,生成满足 $f(x+\Delta x)=l$ 的对抗样本 $x+\Delta x$,其中,$loss_f$ 是 f 的损失函数,c 是一个大于 0 的常数.Goodfellow 等人[16]提出,算法对于对抗样本脆弱性的主要原因正是在于它们的线性本质.以线性模型为例,输入的扰动导致网络的输出变化可以大致表示为 $\omega^T\tilde{x}=\omega^T x+\omega^T\eta$,其中,$\eta$为扰动.对于现有的神经网络模型,它们都表现出比较强的线性特性,由于神经网络的输入是高维数据,因此只需要较小的扰动 η,网络的输出 $\omega^T\tilde{x}$ 在高维度空间下就会发生很大的变化.基于上述分析,Goodfellow 等人提出了一种能够快速产生对抗样本的方法:FGSM.该方法在 MNIST 数据集下对对抗样本的各个方面进行了评估,包括对抗样本在具有权重衰减的网络下的表现、在深度网络下的表现以及在不同类型网络下的表现.这些实验产生了很多结论,比如对抗样本扰动的方向比具体数值更重要;越容易优化的模型越容易产生对抗样本;用对抗样本训练后的网络无法抵御对抗样本的攻击.这些结论在对抗样本的研究上具有重要意义.Xiao 等人[46]提出了基于空间变换的图像对抗样本生成方法.与其他直接操纵图片中每一个像素的值产生对抗样本的方法不同,该方法通过对图像中的像素在空间位置上进行移动,在原图片基础上生成对抗样本.Xiao 等人认为:直接操纵图片中像素的值容易对图片造成肉眼可见的变化,导致图片失真.相比之下,通过移动图片的像素生成对抗样本的过程更加平滑,能够产生感知上更加真实的对抗样本.该方法在 MNIST,CIFAR-10 与 ImageNet 这 3 个数据集上与两个著名的对抗样本生成方法 FGSM 与 C&W 进行了对比.结果表明,该方法生成的对抗样本更难被人察觉.

剩余的工作关注生成对抗样本的其他方面.He 等人[47]提出了一种针对区域分类的对抗样本生成方法,命名为 OptMargin.区域分类是由 Cao 等人提出的一种对抗攻击防御方法[57],该方法在输入样本的临近空间中采样多个样本并输入分类器进行分类,并综合分类结果得到最后的预测.区域分类方法相比只对一个样本输入进行预测的点分类方法能很好地防御多种对抗攻击.OptMargin 考虑了区域分类的情况,通过设计的损失函数成功生成了区域分类无法防御的对抗样本.He 等人基于实验结果证明:OptMargin 在点分类与区域分类两种情况都具有很稳定的攻击效果,生成的样本在达到相同攻击效果的情况下比 C&W 与 FGSM 更难被分辨.

2.2.2 黑盒方法

部分工作在生成对抗样本时只关注 DNN 模型的输入和输出,而不关注网络的内部状态,我们称之为黑盒方法.Brendel 等人[48]提出了一种基于决策的对抗样本生成方法,称为边界攻击.边界攻击指的是以原样本和一个已获得的对抗样本为初始条件,通过对样本空间中模型错误决策的区域(称为对抗区域)与正确决策的区域(称为非对抗区域)之间的边界进行探索,从而寻找到一个能够使模型作出错误决策,且与原样本具有更小差异的新对抗样本.该方法使用 MNIST,CIFAR-10 和 ImageNet 数据集与其他方法(FGSM,DeepFool[58]与 C&W)进行了实验比较.相比其他方法,边界攻击不仅过程简单且具有更少的超参数,而且在扰动大小上表现出了不亚于其他基于梯度下降方法的性能.一些研究者则关注对抗样本的真实性.Zhao 等人[49]提出了一种生成更加自然的对抗样本的方法,他们认为:现有的对抗样本的生成方式是直接在样本的输入空间中进行搜索,导致对抗样本与原样本的差异没有意义,生成的对抗样本是“非自然”的,不能对 DNN 模型的测试产生很大的帮助.Zhao 等人使用 GAN 学习样本在输入空间中的分布,然后在该分布中寻找相应的对抗样本,以此生成更加自然的对抗样本.他们在图片输入和文本输入上展开实验.结果表明,该方法生成的对抗样本对 DNN 的测试更加有帮助.Wicker 等人提出一种特征引导的对抗性样本的鲁棒性测试方法[51],该方法借助尺度不变特征转换算法提取特征,通过双方博弈游戏的方式确定特征和操作像素点,并利用蒙特卡罗树搜索算法逐步探索博弈状态空间来生成对抗性样本.Wicker 在 MNIST 和 CIFAR-10 数据集上展开实验,并证明了该方法在对抗性样本生成方面的有效性和实时性;并基于 Nexar 交通灯识别[59]数据集,进一步证明了该方法可以用来评估神经网络在自动驾驶汽车交通标志识别等安全关键应用中的鲁棒性.

3 测试预言

软件测试预言问题[60]是指软件在测试过程中需要在给定的输入下能够区分出软件正确行为和潜在的错误行为.与其他软件系统的测试一样,深度神经网络测试也需要解决测试预言问题,即如何判断对给定的测试输入,深度神经网络的输出是否符合预期.为了解决该问题,研究人员提出了多种方法.本文将这些方法分为两类:

蜕变测试和差异测试.下面两小节分别介绍该两类方法.

3.1 蜕变测试

蜕变测试是一种解决测试预言问题的常用方法[61−64].它的核心思想是构造蜕变关系,即描述待测系统的测试输入的变化与输出的变化的关系.例如:当测试 sin 函数时,假设给定测试输入为 1,确定 sin(1)的预期输出是十分困难的.然而,sin 函数具有一些数学属性,如 sin(−*x*)=−sin(*x*),可以辅助测试该函数.对于上述给定的测试输入 1,我们可以通过比较−sin(1)和 sin(−1)是否相等来辅助测试 sin 函数.近年来,蜕变测试也被应用到机器学习算法和 DNN 系统的测试中,其中所使用的蜕变关系见表 3.

Table 3 Summary of metamorphic relations in related works of metamorphic testing

表 3 蜕变测试相关工作中的蜕变关系总结

工作	应用领域	使用的蜕变关系
文献[65]	排序系统	1. 对输入加相同值不改变输出 2. 对输入乘相同值不改变输出 3. 对输入小幅度扰动不改变输出 4. 对输入取反后的输出是可预料的 5. 对包含于输入及内的新输入的输出是可预料的 6. 对不包含于输入集内的新输入的输出是不可预料的
文献[66]	图像	1. 对所有训练数据和测试数据中特征的值做仿射变换后,预测结果保持一致 2. 对所有训练数据和测试数据中的标签做一致的打乱后,预测结果保持一致 3. 对所有训练数据和测试数据中的 *m* 个特征进行一致的打乱,预测结果不变 4. 在测试数据中添加无关属性,测试数据的预测结果不变 5. 在分类结果为 l 的测试数据中添加与分类 l 相关度高的属性,分类结果仍为 l 6. 将某数据和其分类结果添加到训练集中,训练出的新分类器的预测结果不变 7. 将训练数据中与预测标签一致的数据重复,训练出的新分类器的预测结果不变 8. 假设某数据分类结果为 l,将训练数据中标签不为 l 的部分数据的标签重设为一个新的类别 *m* 后加入训练集,训练出的新分类器对该数据的分类结果仍为 l 9. 假设某数据分类结果为 l,将训练数据中标签不为 l 的部分数据的标签重设为一个新的类别 *m* 后替换原数据,训练出的新分类器对该数据的分类结果仍为 l 10. 假设某数据分类结果为 l,将训练数据中某些不为 l 的标签的所有数据移出训练集,训练出的新分类器对该数据的分类结果仍为 l 11. 假设某数据分类结果为 l,将训练数据中某些不为 l 的标签的部分数据移出训练集,训练出的新分类器对该数据的分类结果仍为 l
文献[67]	图像	1. 对所有训练数据和测试数据中的标签做一致的打乱后,预测结果保持一致 2. 打乱训练集不影响预测结果 3. 交换图片的 RGB 通道不影响图片的预测结果 4. (只针对 SVM 的线性核函数)对特征进行线性变换不影响预测结果
文献[68]	多领域	1. 交换数据顺序 2. 交换数据中的特征顺序 3. 交换数据中特征的名称(不交换特征的值) 4. 将数据中的类别特征替换为数值型(例如将“男”替换为 1)
文献[69]	图像	1. 添加眼镜不影响人脸识别结果 2. 改变妆容不影响人脸识别结果 3. 改变发色不影响人脸识别结果 4. 改变发型不影响人脸识别结果
文献[70]	图像	1. 修改图片对比度不影响预测结果 2. 修改图片亮度不影响预测结果 3. 修改图片锐度不影响预测结果 4. 对图片进行模糊不影响预测结果 5. 对图片进行小幅度扰乱不影响预测结果 6. 对图片进行仿射变换不影响预测结果

蜕变测试在对机器学习算法的测试中有很多成熟的应用.Murphy 等人[65]对机器学习算法 MartiRank 进行分析,构造了 6 种蜕变关系,来对机器学习算法进行蜕变测试.通过分析 MartiRank 算法(其主要功能为排序),Murphy 等人构造的蜕变关系包括:1) 对输入加相同值不改变输出;2) 对输入乘相同值不改变输出;3) 对输入小幅度扰动不改变输出;4) 对输入取反后的输出可预料的;5) 对包含于输入集内的新输入的输出是可预料的;

6) 对不包含于输入集内的新输入的输出是不可预料的.Murphy 等人发现,该 6 种蜕变关系在其他机器学习算法(包括 SVM-Light 和 PAYL)上也是广泛存在的.Xie 等人[66]提出了一种针对有监督的机器学习分类算法的蜕变测试方法.该方法不仅能够检测待测机器学习算法实现的正确性,还能够判断所使用的机器学习算法的合理性.具体来说,Xie 等人总结出了 11 种蜕变关系.当在蜕变测试过程中发现某一种蜕变关系被违反,他们进一步分析该违反是由于该机器学习算法的实现不正确,还是该蜕变关系对该算法不适用.基于该方法,Xie 等人对 k 近邻算法与朴素贝叶斯算法的 Weka 实现进行了测试,实验阐明了该方法的效果.

蜕变测试也越来越多地被应用到对深度神经网络的测试中.Dwarakanath 等人[67]提出了一种针对图像分类的 DNN 系统的蜕变测试方法.他们针对图像数据设计了多种蜕变关系,然后利用这些蜕变关系对多个图像分类器进行了测试.实验结果表明,该方法可以有效检测出实现这些分类器中的代码缺陷.数据平衡性问题指的是针对输入数据的某个特征的修改会导致网络预测结果发生变化,该问题会导致深度神经网络无法从训练数据中学习到期望的内容.因此,检测训练数据的平衡性也是深度神经网络测试中的重要内容.Sharma 等人[68]提出一种基于蜕变测试的检测 DNN 系统中训练数据平衡性问题的方法,他们针对测试训练数据的平衡性提出了 4 种蜕变关系.Zhu 等人[69]设计了一系列测试图像领域 DNN 系统的蜕变关系,并在人脸识别应用上进行了实验探究.基于所提出的蜕变关系,Zhu 等人使用 CelebA[71]和 PubFig[72]人脸数据集对 4 个成熟的人脸识别系统进行实验,以此来论证该方法的有效性.Braiek 等人[70]结合蜕变测试和神经元覆盖,提出了基于搜索的深度学习系统测试输入生成方法.他们采用了像素层面以及仿射蜕变关系,结合神经元覆盖,高效生成更加多样的图片测试样本.

3.2 差异测试

为了解决测试预言问题,McKeeman 等人[73]提出了差异测试的概念,即相同输入在基于相同规约的多个实现下的输出是相同的.目前,差异测试也被应用于深度神经网络的测试中,用于解决其测试预言问题.

Udeshi 等人[74]提出了 OGMA 测试方法,OGMA 利用神经网络模型的鲁棒性来生成测试输入,并利用差异测试来检测缺陷:如果同一输入在两个功能相同的分类器下得到不同的输出,则该输入揭示了这两个分类器中至少一个分类器的缺陷.Udeshi 等人认为,容易揭错的输入应该是相邻的.因此,他们使用基于语法推导树的方式对输入进行扰动,即对推导树的叶子节点进行替换来生成测试输入,并在 3 个分类器上验证了该方法的有效性.实验结果表明:在特定阈值下,OGMA 生成的输入比随机生成的输入更加有效.

Pham 等人[75]提出了一种针对深度学习框架的测试方法 CRADLE.不同深度学习框架中包含着对大量功能的等价实现,相同的测试输入在不同框架下的相同实现中应具有相同的输出.CRADLE 利用 Keras[76]可以配置加载 Tensorflow[77],Theano[78]和 CNTK[79]等多个后端的特性,对比同一个深度网络模型在不同后端下的表现是否一致来进行缺陷检测.此外,CRADLE 可以对所检测到的不一致进行进一步的缺陷定位,即找到最有可能造成不一致的网络层.

4 深度神经网络测试研究的主要应用领域

图 6 展示了深度神经网络测试研究的主要应用领域,包括图像处理领域、语音识别领域和自然语言处理领域.其中:在图像处理领域的应用最多,主要以对抗性样本生成以及对抗性样本在真实世界中的研究为主;语音处理方面,从针对传统的基于高斯混合模型(Gaussian mixed mode,简称 GMM)语音识别系统的攻击,演化为对现今基于深度神经网络的语音识别系统的攻击;深度神经网络测试在自然语言处理领域的应用主要以文本对抗样本的研究为主.

随着近年来无人驾驶技术的发展,深度神经网络测试在自动驾驶领域的应用也备受关注.针对深度神经网络测试的研究在一定程度上填补了无人驾驶系统测试领域的空白,有助于提高自动驾驶系统的安全性.同时,针对复杂自动驾驶系统的测试也包含了图像和语音等多领域的测试.

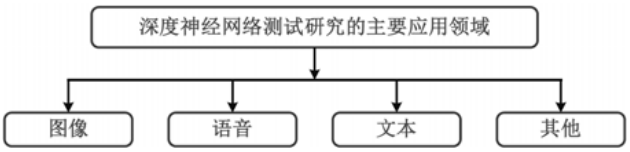

Fig.6 Main application fields of deep neural network testing research

图 6 深度神经网络测试研究主要应用领域

4.1 图像处理上的应用

近些年,研究人员在图像分类领域展开了大量的对抗性样本研究工作,通过相应的对抗性样本生成算法,将生成的图片直接输入到被测模型中,试图欺骗和诱导模型进行错误分类.Kurakin 等人在文献[17]中表明:即便是在真实物理世界场景中,深度学习应用系统也依然容易受到对抗性样本的攻击.Kurakin 等人利用 FGSM,BIM,ILCM 等算法生成对抗性样本,将所生成的对抗性输入打印出来,并使用 Tensorflow Camera Demo App 来模拟真实世界中的目标分类场景.Kurakin 等人基于 Inception v3 模型,在 ImageNet 数据集上展开实验.实验结果证明:现有的部分对抗性样本生成算法同样适用于物理世界中的机器学习系统,并能够成功诱导模型产生错误分类行为.

受到 Kurakin 等人研究工作[17]的启发,Athalye 等人针对三维物体的对抗性样本开展了相关研究工作[80].他们提出:Kurakin 等人没有研究物体在二维空间中的旋转、平移、倾斜和缩放等转换,同时,该方法无法被应用到 3D 物体中.Athalye 等人提出了 3D 物体的转换方法 EOT,在此框架下,基于 3D 打印技术指导三维对抗性样本生成,并在 InceptionV3 模型上展开实验.实验结果表明,对抗性样本是真实世界系统需要关注的问题.

随着人脸识别技术在生活中的普遍应用,针对这类安全关键 DNN 系统的质量和隐私保护等方面的研究尤为重要.Mirjalili 等人将隐私保护扩展到人脸图像识别领域,设计了一种修改人脸图像的技术[81].该方法利用采用梯度下降技术,迭代地向图像中添加轻微扰动,使得性别分类器不易从图像中提取性别特征,从而导致性别分类器做出错误判断,而图像仍可用于人脸识别等用途.

在图像领域,很多通过对抗攻击方法生成的图片样本与原样本可能仅在某些像素上存在微小区别,这些差异难以被摄像头捕捉.因此,在对这些图片进行重拍照后,它们被神经网络产生错误分类的概率会大大降低.Evtimov 等人通过在图片上生成区块遮挡产生对抗样本,并将这一方法应用在交通路牌识别任务中[82].实验表明,该方法产生的图片对抗样本在重拍照后仍保持了相当高的错误分类概率.因此,该方法在现实应用领域具有更高的研究和应用价值.

4.2 语音处理上的应用

语音领域的深度学习技术也同样会受到对抗性样本的攻击.针对此类系统的对抗样本的研究,对于有效防御攻击和提高模型的鲁棒性具有重要意义.

Vaidya 等人[83]利用人类和机器语音识别的差异对语音系统进行攻击.Carlini 等人[84]提出:任何给定的源音频样本都可以通过将噪音隐藏进原始音频的方式,诱使自动语音识别(automatic speech recognition,简称 ASR)系统将音频转录为不同的目标句子.但是,这种白盒的攻击方法仅适用于传统语音识别系统,而非基于 DNN 的 ASR 系统.部分工作[85,86]利用麦克风固有的非线性特性,通过向语音信号中注入被编辑过的超声波语音命令来攻击 Siri 等语音识别系统,从而达到操纵手机甚至汽车导航系统的目的.由于此类方法利用麦克风硬件漏洞而非 DNN 的弱点,即便这种攻击方法完全无法被人类察觉,但依然可以通过使用增强型麦克风来防御攻击.

Yuan 等人[87]提出了一种实用且系统的对抗性攻击方法 CommanderSong,该方法将一组命令嵌入到一首歌曲中,从而合成包含命令信息的并且可远程传播歌曲.其能够对 iFLYTEK 2,Kaldi 等主流 ASR 系统进行黑盒攻击,但对原始音频引入了显著的扰动.针对 CommanderSong 噪音过大的问题,部分工作[88,89]利用心理声学隐藏模型,将对抗性扰动添加到人类无法感知的音频区域生成对抗性样本,降低插入噪声被人类识别的可能性.Xu 等人[90]提出:ASR 系统可能会对用户对话内容进行监控,从而导致用户隐私泄露等安全问题.因此,他们利用 ASR

系统对于对抗样本的敏感性,在 MFCC(mel-frequency cepstral coefficients,简称 MFCC,语音信号的一种特征)生成的阶段注入扰动,提出了针对移动设备恶意 ASR 监控的高性能自适应安全增强方案 HASP.

现阶段,已有一些针对语音情感识别(speech emotion recognition,简称 SER)系统测试及对抗性样本生成的研究.Latif 等人[91]首次提出了针对 SER 系统对抗样本生成的黑盒方法,并提出了基于 GAN 的防御措施以增强 SER 系统的鲁棒性.他们在 IEMOCAP[92]和 FAU-AIBO[93]两个情感语料库上,使用 Demand Noise 数据库[94]中的咖啡馆、会议和电台数据作为插入噪声,对 LSTM 架构的情感分类网络进行攻击.除此之外,还有部分工作[95−97]使用 voice squatting 和 voice masquerading 方法对语音助手及相应第三方应用进行攻击.

4.3 自然语言处理上的应用

Papernot 等人[98]提出一种针对 RNN 文本分类的攻击方法:在输入文本的随机位置选择单词,然后使用投影的 FGSM 算法为单词的嵌入向量添加扰动,以此随机改变其输入特征.通过将被扰动的向量投影到单词嵌入空间中以寻找其最接近的单词向量,从而生成对抗性序列.在电影评论情绪分类任务中,该方法平均在 71 个单词的电影评论中更改 9 个单词,就可以让 RNN 出现 100%错误预测.

图像和音频数据是连续的,对噪声的容忍较大;与其不同的是,文本是离散数据,即便微小的扰动也会使文本发生变化,甚至无法识别.Liang 等人[99]提出:FGSM 算法得到的文本虽然能被模型错误分类,但文本难以被阅读.他们提出了一种新的基于 DNN 的文本分类器的对抗样本的生成方法,并提出了白盒和黑盒两种攻击方案.在实验中,他们基于字符级模型[100]和单词级模型[101]两个代表性的文本分类 DNN 作为攻击目标,在多类数据集 DB-pedia[102]进行实验.实验结果表明,该方法可以有效地进行源/目标的错误分类攻击.

Samanta 等人[103]通过删除或替换文本中的重要单词等手段修改原始样本,产生文本对抗样本,并在 IMDB 电影评论数据集[104]和 Twitter 数据集[105]上进行实验来验证所提方法的有效性.Lei 等人[106]提出了一种对抗样本攻击的通用框架,该方法采用梯度引导的贪心策略,同时具备了梯度引导方法的效率和贪婪方法的能力,相比传统的贪婪算法耗时更少,并使用联合句和单词释义技术保证了文本的原始语义和语法.在自然语言处理领域,序列到序列(seq2seq)模型广泛应用于自动应答、机器翻译等场景,因此部分研究者聚焦于针对 seq2seq 对抗性样本方面的研究.Cheng 等人提出了生成 seq2seq 神经网络模型的对抗样本的方法[107],该方法能够实现非重叠攻击和有针对性的关键词攻击.与 HotFlip[108]不同,该方法提出了一种投影梯度法用来解决离散输入空间问题,因此可以进行有针对性的攻击,而 HotFlip 只关注非目标攻击.

4.4 其他应用

近年来,深度学习被广泛被应用于自动驾驶等安全关键领域.深度学习技术贯穿自动驾驶系统的感知、决策和控制等多个模块.在边界输入条件下,基于深度学习技术的自动驾驶系统做出的错误行为往往会产生灾难性的后果.因此,针对自动驾驶系统的测试,对于提高其可靠性和安全性具有重要意义.

现有部分工作通过产生具有不同语义信息的测试场景,对基于图像的感知模块进行测试.Tian 等人[109]提出一种针对深度神经网络驱动的自动驾驶系统的测试方法 DeepTest,该方法产生能够模拟驾驶场景中的天气信息,同时生成最大化神经网络中激活神经元个数的测试图像,以测试自动驾驶系统在不同场景下的行为.Ramanagopal 等人[110]提出一种针对自动驾驶系统的自动化测试方法,利用未标注数据来识别目标检测器产生的错误行为.他们提出:Zhang 等人[111]提出的 ALERT 框架和 Darfty 等人[112]将空间-时间特征应用到神经网络的方法,都不能直接用于识别没有标记的物体.Ramanagopal 等人研发的系统受益于可靠的置信度估计,因此可以减少目标检测器的误报.在实验中,他们使用 3 个最先进的目标检测网络,即 SSD,Faster R-CNN 和 RRC,在 Sim200k[113]数据集上展开实验.结果表明:该方法可以高效地侦测目标检测器的真实故障,并能够在不同的模拟天气情况下取得较好效果.

在 DNN 测试中广泛使用的具有强大的图像生成能力的 GAN 网络在自动驾驶测试领域也得到应用,Zhang 等人[114]提出一种基于 GAN 的自动驾驶系统蜕变测试和输入验证框架 DeepRoad.该框架采用无监督的方式,自动生成大量精确的驾驶场景测试用例,同样可以达到模拟天气变化的效果.

Tuncali 等人[115]提出了一种基于仿真的自动驾驶系统对抗案例生成框架 Sim-ATAV.该框架提供了一种用于检测自动驾驶系统的闭环特性新算法,并将覆盖阵列应用到基于机器学习的闭环网络物理系统.与 Dreossi 等人[116]提出的使用静态图像识别候选反例,并模拟闭环行为验证安全性的方法不同,Tuncali 等人提出了一种直接在系统的闭环行为上搜索不安全行为的新方法,即采用闭环行为作为损失函数的全局优化器.Tuncali 等人认为:O'Kelly 等人[117]的方法不能在闭环中模拟 ego 车辆系统,只在明确定义的简单车辆动态模型中分析,不能很好地模仿真实行为.Tuncali 等人在 Webots 中搭建了一个特定的驱动场景对 Sim-ATAV 框架进行验证,将场景中的事件用 STL 规范进行描述,使用覆盖阵列测试各离散参数的组合,并使用模拟退火算法寻找边界输入,以达到定位虚拟场景中危险测试用例的效果.

5 深度神经网络测试中的评测数据集

为了更好地支持相关研究的调研及复现,本文对深度神经网络测试领域常用的数据集进行了统计,具体分析结果见表 4.表中列举了图像相关数据集 8 个,文本相关数据集 3 个,语音相关数据集 7 个,自动驾驶测试集 3 个,介绍了相关数据集的简要信息并提供数据集的下载链接.

Table 4 Summary of datasets

表 4 数据集总结

领域	数据集名称	样本规模	类别	下载地址
图像	MNIST	7 万	10	文献[13]
	ImageNet	1 400 万	2 万+	文献[14]
	CIFAR-10	6 万	10	文献[22]
	CIFAR-100	6 万	100	文献[22]
	COCO	33 万+	80	文献[23]
	LSUN	–	–	文献[50]
	CelebA	20 万+	–	文献[71]
	PubFig	5 万+	200	文献[72]
自然语言处理	DBpedia	–	–	文献[102]
	IMDB movie review dataset	5 万	2	文献[104]
	Twitter gender classification dataset	2 万+	3	文献[105]
语音	Fisher	–	–	文献[25]
	LibriSpeech	1 000h	–	文献[26]
	Switchboard	300h	–	文献[27]
	Mozilla common voice dataset	近 1 400h	–	文献[28]
	IEMOCAP	12h	–	文献[92]
	FAU-AIBO	9h	–	文献[93]
	Demand noise database	–	–	文献[94]
自动驾驶	Udacity driving	–	–	文献[15]
	Nexar traffic light recognition	1.8 万+	–	文献[59]
	Sim200k	20 万	–	文献[113]

6 总结和展望

随着深度神经网络被越来越广泛地应用于不同领域,深度神经网络的质量也受到了广泛关注.本文提供了针对深度神经网络测试的综述.具体来说,本文从测试度量指标、测试输入、测试预言、以及应用领域这 4 个方面对该领域进行了详细全面的综述.

尽管目前已经有许多研究围绕着深度神经网络的测试进行展开,但该领域仍然面临着许多挑战.在这里,本文对仍然存在的主要挑战进行概括,以希望未来有更多的研究围绕着这些挑战开展,从而进一步保障深度神经网络的质量.

1. 在深度神经网络的质量保证中,除了深度神经网络的测试之外,深度神经网络的验证也是一种重要手段.现有的深度神经网络验证是把问题归结为一个约束求解问题,即将深度神经网络建模为基于约束

的问题,使用线性规划求解器和 SMT 求解器来进行求解[36,118−120].如何将深度神经网络的测试与验证方法进行有效地结合,从而进一步保证深度神经网络的质量,是一项尚待解决的问题.

2. 深度神经网络的测试过程往往需要较大的计算资源,主要原因包括:1) 测试输入数量较大;2) 对测试输入进行标注的代价较大.如何加速深度神经网络的测试,是一项需要被解决的问题.在传统软件的测试中,测试用例排序、约减和选择是加速软件测试的主要手段.借鉴这些方式,对深度神经网络的测试输入进行排序、约减和选择,是解决高计算开销问题的一个可能的探索方向.
3. 尽管目前有许多深度神经网络的测试方法被提出,但相比于传统软件测试较为系统的方法和流程,深度神经网络的测试的方法和流程仍比较模糊.因此,系统地比较不同深度神经网络测试方法在实际中的测试效果,也是一个值得探索的方向.该方向不仅可以探究不同测试方法之间的关系,还可以了解当前学术研究与实际工业界应用之间的距离.

References:

[1] Hinton GE, Salakhutdinov R. Reducing the dimensionality of data with neural networks. Science, 2006,313(5786):504−507.

[2] Huang X, Kroening D, Kwiatkowska M, *et al*. Safety and trustworthiness of deep neural networks: A survey. arXiv preprint arXiv: 1812.08342, 2018.

[3] Zhang JM, Harman M, Ma L, Liu Y. Machine learning testing: Survey, landscapes and horizons. arXiv:1906.10742, 2019.

[4] Xiang W, Musau P, Wild AA, *et al*. Verification for machine learning, autonomy, and neural networks survey. arXiv preprint arXiv: 1810.01989, 2018.

[5] Pei K, Cao Y, Yang J, *et al*. Deepxplore: Automated whitebox testing of deep learning systems. In: Proc. of the 26th Symp. on Operating Systems Principles. ACM, 2017. 1−18.

[6] Sun Y, Huang X, Kroening D. Testing deep neural networks. arXiv preprint arXiv:1803.04792, 2018.

[7] Ma L, Juefei-Xu F, Zhang FY, *et al*. DeepGauge: Multi-granularity testing criteria for deep learning systems. In: Proc. of the Automated Software Engineering. 2018. 120−131.

[8] Ma L, Juefei-Xu F, Xue M, *et al*. DeepCT: Tomographic combinatorial testing for deep learning systems. In: Proc. of the 2019 IEEE 26th Int'l Conf. on Software Analysis, Evolution and Reengineering (SANER). IEEE, 2019. 614−618.

[9] Wang D, Wang Z, Fang C, *et al*. DeepPath: Path-driven testing criteria for deep neural networks. In: Proc. of the 2019 IEEE Int'l Conf. On Artificial Intelligence Testing (AITest). IEEE, 2019. 119−120.

[10] Tian Y, Zhong Z, Ordonez V, *et al*. Testing deep neural network based image classifiers. arXiv preprint arXiv:1905.07831, 2019.

[11] Du X, Xie X, Li Y, *et al*. Deepcruiser: Automated guided testing for stateful deep learning systems. arXiv preprint arXiv:1812. 05339, 2018.

[12] Kim J, Feldt R, Yoo S. Guiding deep learning system testing using surprise adequacy. In: Proc. of the 41st Int'l Conf. on Software Engineering. IEEE, 2019. 1039−1049.

[13] http://yann.lecun.com/exdb/mnist/

[14] http://www.image-net.org/

[15] https://github.com/udacity/self-driving-car/tree/master/datasets

[16] Goodfellow IJ, Shlens J, Szegedy C, *et al*. Explaining and harnessing adversarial examples. In: Proc. of the Int'l Conf. on Learning Representations. 2015.

[17] Kurakin A, Goodfellow I, Bengio S, *et al*. Adversarial examples in the physical world. In: Proc. of the Int'l Conf. on Learning Representations. 2017.

[18] Papernot N, McDaniel P, Jha S, *et al*. The limitations of deep learning in adversarial settings. In: Proc. of the 2016 IEEE European Symp. on Security and Privacy (EuroS&P). IEEE, 2016. 372−387.

[19] Carlini N, Wagner DA. Towards evaluating the robustness of neural networks. In: Proc. of the IEEE Symp. on Security and Privacy. 2017. 39−57.

[20] IBM. IBM CPLEX mathematical programming solver for linear programming. https://www.ibm.com/analytics/cplex-optimizer

[21] Sekhon J, Fleming C. Towards improved testing for deep learning. In: Proc. of the 41st Int'l Conf. on Software Engineering: New Ideas and Emerging Results. IEEE, 2019. 85−88.

[22] http://www.cs.toronto.edu/~kriz/cifar.html

[23] http://cocodataset.org/

[24] Mozilla's DeepSpeech. 2018. https://github.com/mozilla/DeepSpeech
[25] https://catalog.ldc.upenn.edu/LDC2004T19
[26] http://www.openslr.org/12
[27] https://catalog.ldc.upenn.edu/LDC97S62
[28] Commom voice dataset. https://voice.mozilla.org/en/datasets
[29] Li ZN, Ma XX, Xu C, *et al*. Structural coverage criteria for neural networks could be misleading. In: Proc. of the 41st Int'l Conf. on Software Engineering: New Ideas and Emerging Results. IEEE, 2019. 89–92.
[30] Ruan W, Huang X, Kwiatkowska M. Reachability analysis of deep neural networks with provable guarantees. In: Proc. of the 27th Int'l Joint Conf. on Artificial Intelligence. AAAI, 2018. 2651−2659.
[31] Huang X, Kwiatkowska M, Wang S, *et al*. Safety verification of deep neural networks. In: Proc. of the Int'l Conf. on Computer Aided Verification. Cham: Springer-Verlag, 2017. 3–29.
[32] ÓSearcóid M. Metric Spaces, Springer Undergraduate Mathematics Series. Berlin, New York: Springer-Verlag, 2006. 154−156.
[33] Szegedy C, Zaremba W, Sutskever I, *et al*. Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199, 2013.
[34] Xu H, Mannor S. Robustness and generalization. Machine Learning, 2012,86(3):391−423.
[35] Weng TW, Zhang H, Chen PY, Yi JF, Su D, Gao YP, Hsieh CJ, Daniel L. Evaluating the robustness of neural networks: An extreme value theory approach. In: Proc. of the 6th Int'l Conf. on Learning Representations. 2018.
[36] Katz G, Barrett C, Dill D L, *et al*. Reluplex: An efficient SMT solver for verifying deep neural networks. In: Proc. of the Int'l Conf. on Computer Aided Verification. Cham: Springer-Verlag, 2017. 97−117.
[37] Papernot N, McDaniel P, Wu X, *et al*. Distillation as a defense to adversarial perturbations against deep neural networks. In: Proc. of the 2016 IEEE Symp. on Security and Privacy (SP). IEEE, 2016. 582–597.
[38] Mangal R, Nori AV, Orso A. Robustness of neural networks: A probabilistic and practical approach. In: Proc. of the 41st Int'l Conf. on Software Engineering: New Ideas and Emerging Results. IEEE, 2019. 93–96.
[39] Bastani O, Ioannou Y, Lampropoulos L, *et al*. Measuring neural net robustness with constraints. In: Proc. of the 30th Int'l Conf. on Neural Information Processing Systems. Curran Associates Inc., 2016. 2621−2629.
[40] Cheng CH, Huang CH, Ruess H, *et al*. Towards dependability metrics for neural networks. In: Proc. of the 2018 16th ACM/IEEE Int'l Conf. on Formal Methods and Models for System Design (MEMOCODE). IEEE, 2018. 1−4.
[41] Guo J, Jiang Y, Zhao Y, *et al*. DLFuzz: Differential fuzzing testing of deep learning systems. In: Proc. of the 2018 26th ACM Joint Meeting on European Software Engineering Conf. and Symp. on the Foundations of Software Engineering. ACM, 2018. 739−743.
[42] Odena A, Olsson C, Andersen D, *et al*. TensorFuzz: Debugging neural networks with coverage-guided fuzzing. In: Proc. of the Int'l Conf. on Machine Learning. 2019. 4901−4911.
[43] Xie X, Ma L, Juefei-Xu F, *et al*. Coverage-guided fuzzing for deep neural networks. arXiv preprint arXiv:1809.01266, 2018.
[44] Sun Y, Wu M, Ruan W, *et al*. Concolic testing for deep neural networks. In: Proc. of the 33rd ACM/IEEE Int'l Conf. on Automated Software Engineering. ACM, 2018. 109−119.
[45] Gopinath D, Wang K, Zhang M, *et al*. Symbolic execution for deep neural networks. arXiv preprint arXiv:1807.10439, 2018.
[46] Xiao C, Zhu J, Li B, *et al*. Spatially transformed adversarial examples. In: Proc. of the Int'l Conf. on Learning Representations. 2018.
[47] He W, Li B, Song D, *et al*. Decision boundary analysis of adversarial examples. In: Proc. of the Int'l Conf. on Learning Representations. 2018.
[48] Brendel W, Rauber J, Bethge M. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. arXiv preprint arXiv:1712.04248, 2017.
[49] Zhao Z, Dua D, Singh S. Generating natural adversarial examples. arXiv preprint arXiv:1710.11342, 2017.
[50] http://www.yf.io/p/lsun
[51] Wicker M, Huang X, Kwiatkowska M. Feature-guided black-box safety testing of deep neural networks. In: Proc. of the Int'l Conf. on Tools and Algorithms for the Construction and Analysis of Systems. Cham: Springer-Verlag, 2018. 408−426.
[52] Liang H, Pei X, Jia X, *et al*. Fuzzing: State of the art. IEEE Trans. on Reliability, 2018,67(3):1199−1218.
[53] Jia Y, Harman M. An analysis and survey of the development of mutation testing. IEEE Trans. on Software Engineering, 2010, 37(5):649−678.

[54] Ma L, Zhang F, Sun J, *et al*. Deepmutation: Mutation testing of deep learning systems. In: Proc. of the 2018 IEEE 29th Int'l Symp. on Software Reliability Engineering (ISSRE). IEEE, 2018. 100−111.

[55] Majumdar R, Sen K. Hybrid concolic testing. In: Proc. of the 29th Int'l Conf. on Software Engineering (ICSE 2007). IEEE, 2007. 416−426.

[56] Li ZN, Ma XX, Xu C, *et al*. Boosting operational DNN testing efficiency through conditioning. In: Proc. of the 27th ACM Joint Meeting on European Software Engineering Conf. and Symp. on the Foundations of Software Engineering. ACM, 2019. 499−509.

[57] Cao X, Gong NZ. Mitigating evasion attacks to deep neural networks via region-based classification. In: Proc. of the 33rd Annual Computer Security Applications Conf. ACM, 2017. 278−287.

[58] Moosavi-Dezfooli SM, Fawzi A, Frossard P. Deepfool: A simple and accurate method to fool deep neural networks. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. 2016. 2574−2582.

[59] https://challenge.getnexar.com/challenge-1/

[60] Barr ET, Harman M, McMinn P, *et al*. The oracle problem in software testing: A survey. IEEE Trans. on Software Engineering, 2014,41(5):507−525.

[61] Dong GW, Xu BW, Chen L, *et al*. Survey of metamorphic testing. Journal of Frontiers of Computer Science and Technology, 2009,3(2):130−143 (in Chinese with English abstract).

[62] Feldt R, Poulding S, Clark D, Yoo S. Test set iameter: Quantifying the diversity of sets of test cases. In: Proc. of the IEEE Int'l Conf. on Software Testing, Verification, and Validation (ICST 2016). 2016. 223−233.

[63] Chen TY, Kuo FC, Liu H, *et al*. Metamorphic testing: A review of challenges and opportunities. ACM Computing Surveys (CSUR), 2018,51(1):1−27.

[64] Segura S, Fraser G, Sanchez AB, *et al*. A survey on metamorphic testing. IEEE Trans. on Software Engineering, 2016,42(9): 805−824.

[65] Murphy C, Kaiser G, Hu L, *et al*. Properties of machine learning applications for use in metamorphic testing. In: Proc. of the SEKE 2008. 2008. 867−872.

[66] Xie X, Ho J, Murphy C, *et al*. Application of metamorphic testing to supervised classifiers. In: Proc. of the 2009 9th Int'l Conf. on Quality Software. IEEE, 2009. 135−144.

[67] Dwarakanath A, Ahuja M, Sikand S, *et al*. Identifying implementation bugs in machine learning based image classifiers using metamorphic testing. In: Proc. of the 27th ACM SIGSOFT Int'l Symp. on Software Testing and Analysis. ACM, 2018. 118−128.

[68] Sharma A, Wehrheim H. Testing machine learning algorithms for balanced data usage. In: Proc. of the 2019 12th IEEE Conf. on Software Testing, Validation and Verification (ICST). IEEE, 2019. 125−135.

[69] Zhu H, Liu D, Bayley I, *et al*. Datamorphic testing: A method for testing intelligent applications. In: Proc. of the 2019 IEEE Int'l Conf. on Artificial Intelligence Testing (AITest). IEEE, 2019. 149−156.

[70] Braiek HB, Khomh F. DeepEvolution: A search-based testing approach for deep neural networks. In: Proc. of the 2019 IEEE Int'l Conf. on Software Maintenance and Evolution (ICSME). IEEE, 2019. 454−458.

[71] http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html

[72] http://www.cs.columbia.edu/CAVE/databases/pubfig/

[73] McKeeman WM. Differential testing for software. Digital Technical Journal, 1998,10(1):100−107.

[74] Udeshi S, Chattopadhyay S. Grammar based directed testing of machine learning systems. arXiv preprint arXiv:1902.10027, 2019.

[75] Pham HV, Lutellier T, Qi W, *et al*. CRADLE: Cross-backend validation to detect and localize bugs in deep learning libraries. In: Proc. of the 41st Int'l Conf. on Software Engineering. IEEE, 2019. 1027−1038.

[76] https://keras.io/

[77] https://www.tensorflow.org/

[78] http://deeplearning.net/software/theano/

[79] https://docs.microsoft.com/en-us/cognitive-toolkit/

[80] Athalye A, Engstrom L, Ilyas A, *et al*. Synthesizing robust adversarial examples. In: Proc. of the Int'l Conf. on Machine Learning. 2018. 284−293.

[81] Mirjalili V, Ross A. Soft biometric privacy: Retaining biometric utility of face images while perturbing gender. In: Proc. of the Int'l Joint Conf. on Biometrics. 2017.

[82] Evtimov I, Eykholt K, Fernandes E, Kohno T, Li B, Prakash A, Rahmati A, Song D. Robust physical-world attacks on deep learning models. arXiv preprint arXiv:1707.08945, 2017.

[83] Vaidya T, Zhang Y, Sherr M, *et al*. Cocaine noodles: Exploiting the gap between human and machine speech recognition. In: Proc. of the 9th USENIX Workshop on Offensive Technologies (WOOT 2015). 2015.

[84] Carlini N, Mishra P, Vaidya T, Zhang Y, Sherr M, Shields C, Wagner DA, Zhou W. Hidden voice commands. In: Proc. of the USENIX Security Symp. USENIX, 2016. 513−530.

[85] Song L, Mittal P. Inaudible voice commands. arXiv preprint arXiv:1708.07238, 2017.

[86] Zhang G, Yan C, Ji X, *et al*. Dolphinattack: Inaudible voice commands. In: Proc. of the 2017 ACM SIGSAC Conf. on Computer and Communications Security. ACM, 2017. 103−117.

[87] Yuan X, Chen Y, Zhao Y, *et al*. Commandersong: A systematic approach for practical adversarial voice recognition. In: Proc. of the 27th USENIX Security Symp. (USENIX Security 2018). 2018. 49−64.

[88] Schonherr L, Kohls K, Zeiler S, *et al*. Adversarial attacks against automatic speech recognition systems via psychoacoustic hiding. In: Proc. of the Network and Distributed System Security Symp. 2018.

[89] Qin Y, Carlini N, Cottrell G, *et al*. Imperceptible, robust, and targeted adversarial examples for automatic speech recognition. In: Proc. of the Int'l Conf. on Machine Learning. 2019. 5231−5240.

[90] Xu Z, Yu F, Liu C, *et al*. HASP: A high-performance adaptive mobile security enhancement against malicious speech recognition. arXiv preprint arXiv:1809.01697, 2018.

[91] Latif S, Rana R, Qadir J. Adversarial machine learning and speech emotion recognition: Utilizing generative adversarial networks for robustness. arXiv preprint arXiv:1811.11402, 2018.

[92] https://sail.usc.edu/iemocap/iemocap_release.htm

[93] https://www5.cs.fau.de/de/mitarbeiter/steidl-stefan/fau-aibo-emotion-corpus/

[94] http://www.irisa.fr/metiss/DEMAND/

[95] Kumar D, Paccagnella R, Murley P, *et al*. Skill squatting attacks on amazon alexa. In: Proc. of the 27th USENIX Security Symp. (USENIX Security 2018). 2018. 33−47.

[96] Zhang N, Mi X, Feng X, *et al*. Dangerous skills: Understanding and mitigating security risks of voice-controlled third-party functions on virtual personal assistant systems. In: Proc. of the IEEE Symp. on Security and Privacy 2019. 2019. 1381−1396.

[97] Alepis E, Patsakis C. Monkey says, monkey does: Security and privacy on voice assistants. IEEE Access, 2017,5:17841−17851.

[98] Papernot N, McDaniel P, Swami A, Harang R. Crafting adversarial input sequences for recurrent neural networks. In: Proc. of the Military Communications Conf. (MILCOM 2016). IEEE, 2016. 49−54.

[99] Liang B, Li H, Su M, *et al*. Deep text classification can be fooled. In: Proc. of the 27th Int'l Joint Conf. on Artificial Intelligence. AAAI Press, 2018. 4208−4215.

[100] Zhang X, Zhao JB, LeCun Y. Character-level convolutional networks for text classification. In: Proc. of the Advances in Neural Information Processing Systems. 2015. 649−657.

[101] Kim Y. Convolutional neural networks for sentence classification. In: Proc. of the 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP). Association for Computational Linguistics, 2014. 1746−1751.

[102] https://wiki.dbpedia.org/develop/datasets

[103] Samanta S, Mehta S. Towards crafting text adversarial samples. arXiv preprint arXiv:1707.02812, 2017.

[104] http://ai.stanford.edu/~amaas/data/sentiment/index.html

[105] CloudFlower. Twitter gender classification dataset. 2013. https://www.kaggle.com/crowdflower/twitter-user-gender-cl

[106] Lei Q, Wu L, Chen PY, Dimakis AG, Dhillon IS, Witbrock M. Discrete attacks and submodular optimization with applications to text classification. arXiv preprint arXiv:1812.00151, 2018.

[107] Cheng M, Yi J, Zhang H, Chen PY, Hsieh CJ. Seq2Sick: Evaluating the robustness of sequence-to-sequence models with adversarial examples. arXiv preprint arXiv:1803.01128, 2018.

[108] Ebrahimi J, Rao A, Lowd D, *et al*. HotFlip: White-box adversarial examples for text classification. In: Proc. of the 56th Annual Meeting of the Association for Computational Linguistics, Vol.2. 2018. 31−36.

[109] Tian Y, Pei K, Jana S, *et al*. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In: Proc. of the 40th Int'l Conf. on Software Engineering. ACM, 2018. 303−314.

[110] Ramanagopal MS, Anderson C, Vasudevan R, *et al*. Failing to learn: Autonomously identifying perception failures for self-driving cars. IEEE Robotics and Automation Letters, 2018,3(4):3860−3867.

[111] Zhang P, Wang J, Farhadi A, Hebert M, Parikh D. Predicting failures of vision systems. In: Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition. 2014. 3566−3573.

[112] Daftry S, Zeng S, Bagnell JA, Hebert M. Introspective perception: Learning to predict failures in vision systems. In: Proc. of the 2016 IEEE/RSJ Int'l Conf. on Intelligent Robots and Systems (IROS). IEEE, 2016. 1743−1750.

[113] Johnson-Roberson M, Barto C, Mehta R, *et al*. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In: Proc. of the 2017 IEEE Int'l Conf. on Robotics and Automation (ICRA). IEEE, 2017. 746−753.

[114] Zhang M, Zhang Y, Zhang L, *et al*. Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems. In: Proc. of the 33rd ACM/IEEE Int'l Conf. on Automated Software Engineering. ACM, 2018. 132−142.

[115] Tuncali CE, Fainekos G, Ito H, *et al*. Simulation-based adversarial test generation for autonomous vehicles with machine learning components. In: Proc. of the 2018 IEEE Intelligent Vehicles Symp. (IV). IEEE, 2018. 1555−1562.

[116] Dreossi T, Donze A, Seshia SA. Compositional falsication of cyber-physical systems with machine learning components. In: Proc. of the NASA Formal Methods (NFM). LNCS 10227, Springer-Verlag, 2017. 357−372.

[117] O'Kelly M, Abbas H, Mangharam R. Computer-aided design for safe autonomous vehicles. In: Proc. of the 2017 Resilience Week (RWS). 2017.

[118] Wang S, Pei K, Whitehouse J, *et al*. Formal security analysis of neural networks using symbolic intervals. In: Proc. of the 27th USENIX Security Symp. (USENIX Security 2018). 2018. 1599−1614.

[119] Dutta S, Jha S, Sanakaranarayanan S, Tiwari A. Output range analysis for deep neural networks. arXiv preprint arXiv:1709.09130, 2017.

[120] Dutta S, Jha S, Sankaranarayanan S, Tiwari A. Learning and verification of feedback control systems using feedforward neural networks. In: Proc. of the IFAC Conf. on Analysis and Design of Hybrid Systems (ADHS). 2018.

附中文参考文献:

[61] 董国伟,徐宝文,陈林,等.蜕变测试技术综述.计算机科学与探索,2009,3(2):130−143.

王赞(1979－),男,江苏泗洪人,博士,副教授,CCF 专业会员,主要研究领域为软件测试,机器学习.

张栋迪(1994－),男,硕士,主要研究领域为程序分析.

闫明(1996－),男,硕士生,主要研究领域为软件测试.

吴卓(1996－),女,硕士生,CCF 学生会员,主要研究领域为软件测试.

刘爽(1987－),女,博士,副教授,CCF 专业会员,主要研究领域为隐私保护,异常检测,软件测试.

陈翔(1980－),男,博士,副教授,CCF 高级会员,主要研究领域为软件缺陷预测,软件缺陷定位,回归测试和组合测试.

陈俊洁(1992－),男,博士,副教授,CCF 专业会员,主要研究领域为软件分析与测试.