深度学习模型白盒测试技术研究

摘要

我们正在步入“深度学习”的时代，深度学习正进入我们的日常生活的方方面面。相比于传统的机器学习算法，深度学习能够更完美地利用海量的数据帮助机器做出更加智能与正确的决定。然而，随着深度学习在一系列领域开展应用，特别是一些安全质量攸关的领域如自动驾驶、人脸识别、机器人和视频监控等，对于深度学习模型的质量评估正成为学术界和工业界研究的热点。

深度学习模型遵循数据驱动的范式，即其内在逻辑/结构是通过数据训练构成的，而非像传统软件那样由预先构建的需求规约/概要设计/详细设计来界定，这导致无法准确地描述深度学习模型的内在逻辑/结构。对于其内在逻辑/结构的验证一般通过深度学习模型的黑盒测试来侦测模型的缺陷，即判断预测的标记与实际标记的是否匹配。已有研究者提出一些白盒测试方法来实现对于深度学习模型的测试，但已有的白盒测试方法仍未能解释缺陷的出现与其内在逻辑/结构的关联性，这也导致其测试的质量难以进行评估。

本研究旨在收集与整理当前研究者提出的深度学习模型白盒测试的度量指标，并结合已有成果提出自己的想法，以期为该领域的研究人员提供参考。

Abstract

We have entered the era of "deep learning", deep learning is infiltrating many aspects of our daily lives. Compared with traditional machine learning algorithms, deep learning can help machines to make smarter and more accurate decisions by using large amounts of data more perfectly. However, with the application of deep learning in a series of fields, especially some areas where safety and quality are at stake, such as self-driving, face recognition, robots and video surveillance, the quality evaluation of deep learning models is becoming a hot topic in academia and industry.

Deep learning models follow a data-driven paradigm, which means the formation of its internal logic/structure is through data training rather than being defined by pre-built requirements protocols or detailed designs, as traditional software does, which makes it impossible to accurately describe the inherent logic/structure of deep learning models. For the validation of its internal logic/structure, we usually detect the defects of the model through the black-box testing of the deep learning models, in order to determine whether the predicted labels match the actual labels. Some researchers have proposed some white-box testing methods to test deep learning models, but the existing white-box testing methods have not been able to explain the correlation between defects and their internal logic/structure, which also makes it difficult to evaluate the quality of them.

The purpose of this study is to collect and collate the measurements of the white-box testing of the deep learning models proposed by the current researchers, and to put forward my ideas in the light of the existing achievements, in order to provide reference for researchers in this field.

关键词：深度神经网络；测试覆盖；测试用例质量度量

1. 引言

深度学习是机器学习的一个分支，其基本结构是神经网络。神经网络是由一系列神经元通过加权边连接形成的网络，模拟大脑神经网络处理、记忆信息的方式进行信息的处理，其中每个神经元代表一个激活函数。深度学习应用了深度神经网络（DNN），它使用多层非线性处理单元进行特征的提取和转换。典型的深度学习算法通常应用一些广泛使用的神经网络结构，例如卷积神经网络（CNN）和循环神经网络（RNN）等。

与普通神经网络不同的是，深度学习网络通常层数很多，分为一个输入层、一个输出层和若干中间的隐含层。样本从输入层输入，通过各个隐含层中的神经元将输入样本的特征抽象到另一个维度空间，使得这些特征能够更好地进行线性划分，再通过输出层输出结果。

深度学习与普通的机器学习一样，需要将数据按照一定的比例划分为训练集、验证集和测试集，基于训练集的数据训练模型，利用验证集进行超参数的调整和过拟合的监控，再利用测试集评价模型的表现及泛化能力。

深度学习在许多领域取得了很大的成就。DeepMind 利用自回归的完全卷积 WaveNet 模型在音频的生成上有了重大突破。Google 公司将 RNN 融入机器翻译，将机器跟人类翻译准确率的差距缩小了 55%-85%。同时，深度学习在图像识别、物理识别等方面也有着重大的意义和较好的表现，十分具有发展潜力及研究价值。

然而，虽然深度学习的应用已经深入到生活的方方面面，但是对深度学习系统的测试研究仍处在早期阶段，一些初步研究工作的重点在于准确性和神经元覆盖，如 DeepXplore，DeepGauge 和 DeepCover，但其是否具有充分性仍然有待研究。深度学习测试，可谓是当下一个十分重要的研究课题。

2. 机器学习测试概述

深度学习本身也为一种机器学习的模式，所以在这里给出一般机器学习的测试方法，并在此基础上挖掘深度学习测试可行的方式。首先给出机器学习中漏洞（ML Bug）和测试（ML Testing）的定义。机器学习漏洞（ML Bug）指机器学习项目中任何导致已有条件和所需条件不一致的缺陷，机器学习测试（ML Testing）指任何揭示机器学习漏洞的操作。在机器学习中的测试包括三个部分：测试工作流、测试属性、测试组件。下面将逐一说明。

2.1 工作流

第一部分是测试工作流。测试工作流是关于如何用不同的测试操作进行机器学习测试的部分，分为离线测试和在线测试。如图 1 所示，上半部分是离线测试，下半部分是在线测试。在离线测试中，开发者先进行需求分析，确定了整个机器学习系统的说明书，同时规划了测试的流程。之后，开发者从已有的数据中抽取部分作为测试用例，或者基于特定的目的生成测试用例，然后生成测试预言（类似于断言，可以用来检查输出是否正确）。在这些测试的准备做完之后，开发者执行测试。测

试过程包含用训练集数据建立模型或针对测试集数据运行已经建立的模型，同时查看测试预言是否被违背。执行完测试之后，开发者使用评价指标来评判测试的质量。但是离线测试缺少实际应用环境，也缺少用户行为的数据收集过程。在线测试就弥补了这些缺点。用户被分成不同的组来实施控制环境，判断哪个模型性能更优。例如 A/B 模型就是在线测试的一个典型模型，抽样的用户会被分成两组，分别使用新、旧两种机器学习模型。

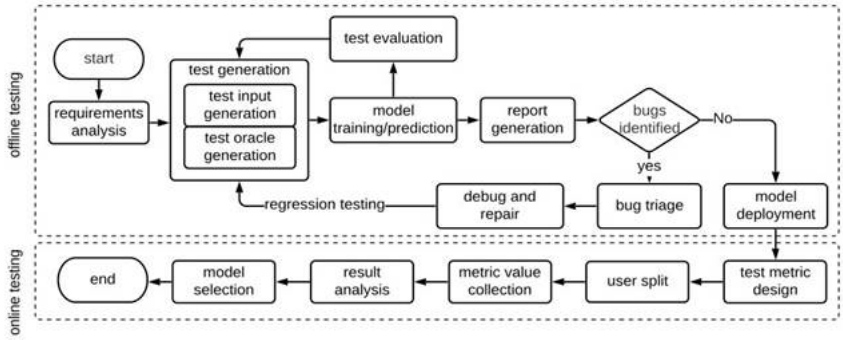

Figure 1 Idealised Workflow of ML testing

2.2 测试属性

第二部分是测试属性。测试属性指为什么要进行机器学习测试，即对于一个已经训练好的模型，机器学习测试需要保证哪些条件。测试属性分为功能性属性（正确性等）和非功能性属性（鲁棒性等）。

例如，正确性度量机器学习系统得到正确结果的能力。设 D 是未知数据的分布，x 是属于 D 的数据，h 表示待测的机器学习模型，h(x)是 x 的预测结果，c(x)是 x 的真实值。模型正确性 E(h)是 h(x)和 c(x)相等的概率：

$$E(h) = Pr_{x\sim D}[h(x) = c(x)]$$

鲁棒性度量机器学习系统在非法输入和高压力环境下正确运行的能力。设 S 是一个机器学习系统。设 E(S)是 S 的正确性，δ(S)是对任何机器学习组件（数据、程序、框架）具有扰动的机器学习系统。机器学习系统的鲁棒性是对 E(S)和 E(δ(S))之间的差异。

$$r = E(S) - E(\delta(S))$$

2.3 测试组件

第三部分是测试组件。机器学习模型需要数据、程序、框架多个组件进行交互，而这些组件都有可能发生错误。而且错误的传播在机器学习模型中尤为严重，因为机器学习模型中三个组件之间的联系比其他传统软件更紧密。在数据组件上的错误检查主要是检查数据是否足够用于训练和测试、是否能代表未来数据、是否包含过多噪声、训练集和测试集是否偏差过大等。在程序组件上的错误检查包括开发者设

计或从框架中选择的算法，以及开发者为实现算法而编写的实际代码。在框架组件上的错误检查主要是检测机器学习框架是否会导致最终系统出现问题。

3. 已有白盒测试方法介绍

类似于传统软件测试，深度学习模型的白盒测试技术领域里，许多研究者也提出了一些关于测试用例覆盖情况的度量指标，然而这种覆盖指标的度量与传统软件测试很不一样，传统软件测试关注的是测试用例对于路径的覆盖情况，即看一个测试用例是否有覆盖到不同的程序语句，但是在深度学习模型里，程序的构成基本上都是在进行一系列的运算，而每次运算的步骤中也几乎会涉及到所有的计算语句，所以采用程序语句覆盖的思想在深度学习模型里是几乎无效的，因此，考虑到深度学习模型模拟的是人体的神经元运作的过程，许多研究者提出了关于神经元覆盖的相关度量指标。在这一节里对一些主要的研究成果进行收集与整理，并在后文中提出自己对于改进的一些想法。

3.1 神经元级别覆盖

该研究指标的思想为考虑在训练过程中每一个神经元的输出情况，对于已经训练好的神经网络，将所有训练数据再次输入进其中，并记录下每个神经元的输出情况，假设共有N个神经元，对于第i的神经元，其输出的最大值为max_i，输出的最小值为min_i，接下来采用集中不同的方式对测试用例的质量进行度量。

3.1.1 k-multisection Neuron Coverage

在这种度量指标下，考虑测试用例对神经元输出值的覆盖情况，即考虑所有的测试用例输入神经网络后是否能够尽可能地将每个神经元训练中的输出情况都覆盖到。

考虑对第i的神经元的情况，其在训练中的输出范围为$[min_i, max_i]$，考虑对这个连续的区间进行离散化，在其中插入$k-1$个分隔点从而将这段区间分成k的小区间，其中第j个区间为

$$[min_i + (j-1)\frac{max_i - min_i}{k}, min_i + j\frac{max_i - min_i}{k}]$$

对于每个测试用例，我们计算其在计算过程中覆盖到了每个神经元的小区间的情况，然后进行汇总，计算每个神经元在所有测试用例都运行过后其k个小区间被覆盖的情况，假设第i个神经元被覆盖的小区间个数为d_i，那么该测试用例集合的覆盖性度量计算公式为：

$$KMNCov = \frac{\sum d_i}{N \cdot k}$$

3.1.2 Neuron Boundary Coverage

在这种度量指标下，考虑测试用例是否能够覆盖到测试数据没有涵盖到的部分，也就是说，对于第i个神经元，考虑测试用例集能否覆盖到

$$(-\infin,min_i),(max_i,\infin)$$

这两个区间，同样记d_i为测试用例集覆盖第i的神经元中这两个区间的个数，那么该测试用例集合的覆盖性度量计算公式为：

$$NBCov = \frac{\sum d_i}{2 \cdot N}$$

3.1.3 Strong Neuron Activation Coverage

该计算指标类似于 3.1.2 中提出的 Neuron Boundary Coverage，只是将第i个神经元考虑的覆盖区间从

$$(-\infty, min_i), (max_i, +\infty)$$

改变为

$$(max_i, +\infty)$$

其认为神经元的输出值越高，越能代表该神经元被激活了，那么该测试就越有效。

3.2 层级别覆盖

该研究指标不要求考虑每一个单独的神经元都要覆盖到，而是以层为单位进行考量，假设神经网络共有s层，对于第i层，共有j个神经元，假设对第x个神经元来说，对于训练数据而言其输出的最大值为max_x，将这些神经元输出的最大值按照从大到小的顺序排列，取其中前k个神经元$N_{i,1}, N_{i,2}, \ldots, N_{i,k}$进行考虑。

3.2.2 Top-k Neuron Coverage

对于测试用例集，收集其在每一层的$top-k$个神经元上的输出值，如果输出值大于该神经元在测试集上的输出值，那么就认为该神经元被覆盖到，假设第i层被覆盖到的神经元的个数为d_i，那么该测试用例集合的覆盖性度量计算公式为：

$$TKNCov = \frac{\sum d_i}{s \cdot k}$$

3.2.1 Top-k Neuron Patterns

在该指标下，考虑每一层的$top-k$个神经元的组合情况的覆盖，假设对于第i层来说，其$top-k$个神经元分别为$N_{i,1}, N_{i,2}, \ldots, N_{i,k}$，对于每一个神经元来说，测试用例在其上的输出值大于训练数据在其上的输出值算是一次激活，考虑该层上被激活的神经元之间的组合，共有2^k种情况，即没有神经元被激活，1 个被激活，2

个被激活，...，k 个被激活的情况，假设测试用例覆盖到2^k种情况中的d_i中，那么该测试用例集合的覆盖性度量计算公式为：

$$TKNPatCov = \frac{\sum d_i}{N \cdot 2^k}$$

4. 现存问题

深度学习模型遵循数据驱动的范式，即其内在逻辑/结构是通过数据训练构成的，而非像传统软件那样由预先构建的需求规约/概要设计/详细设计来界定。这导致无法准确地描述深度学习模型的内在逻辑/结构。对于其内在逻辑/结构的验证一般通过深度学习模型的测试来侦测模型的缺陷，即预测的标记与实际标记的不匹配。

已有研究者提出一些白盒测试方法来实现对于深度学习模型的测试，然而已有的白盒测试方法仍未能解决深度学习模型缺陷的出现与其内在逻辑/结构的关联性，以及测试的充分性准则仍未能有效验证。

现有的白盒测试技术积极地利用神经网络的内部结构：人们选择内部神经元，计算所选元输出的梯度并通过在增加输出值的方向上向原始测试用例添加梯度来生成新的测试用例。与黑盒与灰盒测试相比，白盒方法因此可能获得更高的覆盖率并发现更多的缺陷。这样的白盒方法的一个关键成分就是神经选择策略。为了确定突变的方向，白盒方法首先选择一组内部神经元，然后计算它们的梯度。因此，白盒测试的最终效果取决于第一步神经元的选择。

许多启发式神经元选择策略被提出以在有限的时间预算内实现最大化覆盖。例如，DeepXplore 使用随机选择未激活神经元的策略。DLFuzz 则提出了 4 种不同的神经元选择策略。而使用固定的神经元选择策略则是此类白盒方法的一个主要限制，这导致现有方法的性能在不同的神经网络模型和覆盖率度量中是不一致的。例如，先进的 DLFuzz 在测试具有 Top-k 神经元覆盖的 LeNet-5 时表现良好，然而对于带有 TKNC 的 ResNet-50 而言，DLFuzz 甚至不如单纯的随机方法。

上述问题一直是当前研究关注的热点但仍未有突破。因此，新的白盒研究仍有待开展。

5. 我的研究想法

5.1 改进覆盖准则

5.1.1 分类别的 top-k 覆盖

我认为在 3.2.2 中提到的$Top-k \space Neuron \space Coverage$方法的基础上，应该分类进行考虑。原有的方法认为神经元输出值越高，其被激活的程度也就越高，但这统计方法是对所有的训练数据而言，而这些数据是来源于不同的类别的。就模拟人这一角度来说，不同的神经在识别不同类别或者完成不同任务时发挥着不同的作用，不能一概而论。

在训练的时候，我认为应该按照不同类别的训练数据进行神经元信息的采集，对每个类别的数据都计算出相应的$top-k$个神经元，然后对测试数据也进行分类后计算每个类别的覆盖情况，最终要求每个类别的测试样例需要覆盖到该类别的$top-k$个神经元。这样，神经元的覆盖应当更加准确，同时也能够知道哪一个类别的测试用例还不够，以进行有针对性的测试数据补充，节约成本。

5.2 寻找测试边界值

5.2.1 利用深度神经网络的输出

该想法来源于论文 DeepGini 中提出的方法，我认为神经网络的边界值指的就是在分类任务中结果比较模糊的那些样本，而模糊指的就是输出的结果中存在分到两个类别或以上类别的概率较为接近的情况。利用这种思想，我们可以寻找到当前神经网络中的一些边界值，通过对这些边界值重新训练，或者是通过对这些边界值进行样本的扩增，我认为是可以较好地加强神经网络性能的。

5.2.2 利用内部神经元的激活情况

这个想法是考虑不同类别的样本在相同神经元上应该表现出不同的状况，具体体现在神经元的输出上，所以我设想将训练样本在所有神经元$N_1, N_2, N3, \ldots, Nn$上的输出值记录成一个向量$[x_1, x_2, \ldots, x_n]$，然后对于每一个测试样本y，我们也同样获得一个向量$[y_1, y_2, y_3, \ldots, y_n]$，利用$KNN$的思想判断这个样本与哪个训练样本的分类最为接近，如果恰好与y所对应的样本接近，那么y看成是一个普通的测试样本，如果不是的话，那么我们可以将其看作是一个测试的边界值，应当对其进行重新训练或者进行扩增的操作。

5.3 判断深度神经网络是否有效

5.3.1 利用各层输出的距离

我认为神经网络的本质是将输入数据进行多次线性变化及非线性变换后映射到一个可分类的空间中进行任务的实现，所以在逐层递进的过程中，我认为相同类别的数据应当趋向于聚集在一个更小的区域内，基于这个思想，我认为可以在训练过程中对神经网络的训练是否有效进行一定的衡量。

我们可以基于距离的思想来完成这件事情，对于所有训练数据而言，例如对于一个样本x_0，假设其经过k层之后变换得到$x_{0,k}$，对于同一类别的数据而言，我们可以采用两种方式衡量其是否趋于聚集在同一区域，一是可以计算样本之间的最大距离，二是计算样本分布的直径，若这两个值在神经网络向前传播的过程中变得越来越小，我们有理由认为神经网络将他们确实是看做了同一类的事物。除此之外，我们还要计算不同类别样本之间的距离，可以考虑三种度量手段，一是平均距离，二是最大距离，三是最小距离，倘若这三者在向前传播的过程中变得越来越大，那么就可以认为神经网络将不同类别的事物能够做到基本正确的分离。结合这两者，我认为可以在一定程度上衡量神经网络的训练是否真正有效。

5.3.2 结合其他机器学习

除了利用距离来度量之外，我认为还可以结合其他的机器学习模型来辅助进行判断神经网络的训练是否有效。可以考虑将每一层的输入都作为新的样本，将这些样本输入到其他的机器学习模型如 SVM，DecisionTree 中，计算其他模型在完成相同任务中的效果，若每深一层得到的样本在其他机器学习中能够得到更好的效果，那么就可以认为深度学习模型的空间变换起到了效果。

6. 总结与展望

随着深度神经网络越来越广泛地应用于不同领域，对于其质量的要求也变得越来越高，虽然能够利用传统的黑盒测试对其质量进行初步判定，但是若没有深入到其内部对其内在的逻辑/结构进行测试，终究是不充分的，难以给人信服的说法。现有的白盒测试技术虽然提出了很多覆盖性的准则，也能够对深度神经网络的能力进行强化，但是其更多的是借鉴了传统软件测试的思想与方法，但这种方法对于数据驱动为主体的深度神经网络是否适用仍然是未被证明的，所以我认为在未来的发展中，有以下这些挑战等待着该领域的研究者们进行攻克：

1. 白盒测试的可解释性

   深度神经网络虽然是依靠数据的训练构建起来的，但其内部也并不是杂乱无章的，每一层的神经元通过线性组合构成下一层神经元的输入，那么在线性组合的过程中，我们是否能够寻找到他们之间的关联？例如对一个预测借款人信用值的神经网络来说，若输入层为（银行卡 1 余额，银行卡 2 余额，年龄，工资），那么如果在第一个隐藏层的某个神经元的输入中银行卡 1 余额和银行卡 2 余额所占的权重较高，那么我们可以认为该神经元衡量的是该借款人现在所拥有的金钱数目，以此我们就可以对我们深度神经网络的工作流程进行粗略的解释。倘若我们能够对每一层之间的数据变化进行一定合理地解释，那么我们就更有自信说明我们的深度神经网络是能够正常工作的。

2. 寻找测试充分性准则

   该问题是所有软件测试都面临的一个问题，在计算资源有限的环境下，我们不可能穷尽所有的测试用例，也就意味着我们需要寻找一个合适的充分性准则用来判断我们的测试是否可以终止。对于深度神经网络来说，倘若对其进行了一次修改就要重新运行一次所有的测试用例，那么这些所需要的代价将是巨大的，所以对该问题的研究具有巨大的意义。我认为对于该问题的解答可以借鉴传统软件测试的思想，例如可以考虑对每个测试用例的激活的神经元进行记录，倘若那些神经元在经过重新训练之后参数值被进行了大幅的修改，那么我们的测试就需要覆盖到这些被修改的神经元或者是与这些神经元有关的测试用例我们都需要进行运行等等。倘若测试充分性准则的研究有了较大的进展，也就意味着我们有了进行测试用例选择、排序、约减的手段，也就能够使得不同的测试技术能够在较短时间内完成实践，这将极大地推动机器学习测试技术的发展。

3. 拓宽测试领域

现有的测试研究大都集中在有监督的机器学习任务上，然而我们知道机器学习还有两个重要重要的分支分别是无监督的机器学习和强化学习，这些同样是两个不可忽视的方面。现在这两个分支研究较少的主要原因是测试预言的问题，要解决这两个问题，我们或许要从蜕变与差分的角度来思考，这两种方法可以帮助我们在没有预先知道测试预言的情况下完成对测试的准确性的判定，相关方面的工作如何开展仍然是有待更进一步研究的。

期待在不久的将来能够有更多的对深度学习测试技术的研究，这不仅是软件测试领域的一大难题，更是机器学习进入社会为人所接受的一大要点，倘若做好了这点，我们将能够更好更安心地利用人工智能技术为我们服务。

7. 参考文献

[1] *Nidhra S, Dondeti J. Black box and white box testing techniques-a literature review[J]. International Journal of Embedded Systems and Applications (IJESA), 2012, 2(2): 29-50. DOI:10.5121/ijesa.2012.2204*

[2] *Farzan A, Holzer A, Veith H. Perspectives on White-Box Testing: Coverage, Concurrency, and Concolic Execution[C]//2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST). IEEE, 2015: 1-11. DOI: 10.1109/ICST.2015.7102600*

[3] *Seokhyun Lee, Sooyoung Cha, Dain Lee, and Hakjoo Oh. 2020. Effective white-box testing of deep neural networks with adaptive neuron-selection strategy. In Proceedings of the 29th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2020). Association for Computing Machinery, New York, NY, USA, 165–176. DOI:https://doi.org/10.1145/3395363.3397346**.*

[4] *Lei Ma, Felix Juefei-Xu, Fuyuan Zhang, Jiyuan Sun, Minhui Xue, Bo Li, Chunyang Chen, Ting Su, Li Li, Yang Liu, Jianjun Zhao, and Yadong Wang. 2018. DeepGauge: multi-granularity testing criteria for deep learning systems. In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering (ASE 2018). Association for Computing Machinery, New York, NY, USA, 120–131. DOI:https://doi.org/10.1145/3238147.3238202*

[5] *Kexin Pei, Yinzhi Cao, Junfeng Yang, and Suman Jana. 2017. DeepXplore: Automated Whitebox Testing of Deep Learning Systems. In Proceedings of the 26th Symposium on Operating Systems Principles (SOSP '17). Association for Computing Machinery, New York, NY, USA, 1–18. DOI:https://doi.org/10.1145/3132747.3132785*

[6] *J. Kim, R. Feldt and S. Yoo, "Guiding Deep Learning System Testing Using Surprise Adequacy," \2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, Montreal, QC, Canada, 2019, pp. 1039-1049, DOI: 10.1109/ICSE.2019.00108.**

[7] *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. DOI:https://doi.org/10.1145/3395363.3397357*

[8] *Weijun Shen, Yanhui Li, Lin Chen, Yuanlei Han, Yuming Zhou, Baowen Xu. 2020. Multiple-Boundary Clustering and Prioritization to Promote Neural Network Retraining.*

[9] *Shen W, Li Y, Han Y, et al. Boundary sampling to boost mutation testing for deep learning models[J]. Information and Software Technology, 2020: 106413. DOI:https://doi.org/10.1016/j.infsof.2020.106413*

[10] *J. M. Zhang, M. Harman, L. Ma and Y. Liu, "Machine Learning Testing: Survey, Landscapes and Horizons," in IEEE Transactions on Software Engineering**, DOI: 10.1109/TSE.2019.2962027.*

[11] *Li Z, Ma X, Xu C, et al. Operational calibration: Debugging confidence errors for DNNs in the field[C]//Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. 2020: 901-913. DOI:https://doi.org/10.1145/3368089.3409696*