计 算 机 研 究 与 发 展
Journal of Computer Research and Development
DOI: 10.7544/issn1000-1239.202220803
61(1): 209−220, 2024

基于可解释性分析的深度神经网络优化方法

吴欢欢[1] 谢瑞麟[1] 乔源心[1] 陈 翔[2] 崔展齐[1]
[1](北京信息科技大学计算机学院 北京 100101)
[2](南通大学信息科学技术学院 江苏南通 226019)
(wuhuanhuan@bistu.edu.cn)

Optimizing Deep Neural Network Based on Interpretability Analysis

Wu Huanhuan[1], Xie Ruilin[1], Qiao Yuanxin[1], Chen Xiang[2], and Cui Zhanqi[1]
[1](*School of Computer Science*, *Beijing Information Science and Technology University*, *Beijing* 100101)
[2](*School of Information Science and Technology*, *Nantong University*, *Nantong*, *Jiangsu* 226019)

Abstract In recent years, deep neural networks (DNN) have been widely used in many fields, even replacing human to make decisions in some safety-critical systems, such as autonomous driving and smart healthcare, which requires higher reliability of DNN. It is difficult to understand the internal prediction mechanism and debug because of the complex multi-layer nonlinear network structure of DNN. The existing DNN debugging work mainly improves the performance by adjusting the parameters and augments the training set to optimize DNN. However, it is difficult to control the modification range of adjusting parameters directly, and probably make the model lose the ability of fitting the training set. And unguided augmentation of training set will dramatically increase training costs. To address this problem, a DNN optimization method OptDIA (optimizing DNN based on interpretability analysis) is proposed. Interpretability analysis is conducted on the training process and the decision-making behavior of DNN. According to the interpretability analysis results, the original training data is split into different partitions to evaluate their influences on decision-making results of DNN. After that, the partitions of original training data are transformed with different probabilities to generate new training data, which are used to retrain DNN to improve the performance of the model. The experiments on nine DNNs trained by three datasets shows that OptDIA can improve the accuracy of DNNs by 0.39% to 2.15% and *F*1-*score* of DNNs by 0.11% to 2.03%.

Key words deep neural network; DNN optimization; DNN defect; DNN performance; interpretability analysis

摘 要 近年来,深度神经网络(deep neural network, DNN)在众多领域取得了广泛应用,甚至在安全关键系统中已经可以代替人类作出决策,如自动驾驶和智慧医疗等,这对DNN的可靠性提出了更高的要求.由于DNN具有复杂的多层非线性网络结构,难以理解其内部预测机制,也很难对其进行调试.现有的DNN调试工作主要通过修改DNN的参数和扩增数据集提升DNN性能,以达到优化的目的.然而直接调整参数难以控制修改的幅度,甚至使模型失去对训练数据的拟合能力;而无指导地扩增训练数据则会大幅增加训练成本.针对此问题,提出了一种基于可解释性分析的DNN优化方法(optimizing DNN based on

收稿日期: 2022-09-14; **修回日期**: 2023-02-22
基金项目: 江苏省前沿引领技术基础研究专项 (BK202002001); 国家自然科学基金项目 (61702041); 北京信息科技大学“勤信人才”培育计划项目 (QXTCP C201906)
This work was supported by the Jiangsu Provincial Frontier Leading Technology Fundamental Research Project (BK202002001), the National Natural Science Foundation of China (61702041), and the Beijing Information Science and Technology University "Qin-Xin Talent" Cultivation Project (QXTCP C201906).
通信作者: 崔展齐(czq@bistu.edu.cn)

interpretability analysis, OptDIA). 对 DNN 的训练过程及决策行为进行解释分析, 根据解释分析结果, 将原始数据中对 DNN 决策行为产生不同程度影响的部分以不同概率进行数据变换以生成新训练数据, 并重训练 DNN, 以提升模型性能达到优化 DNN 的目的. 在使用 3 个数据集训练的 9 个 DNN 模型上的实验结果表明, OptDIA 可以将 DNN 的准确率提升 0.39~2.15 个百分点, *F*1-*score* 提升 0.11~2.03 个百分点.

关键词 深度神经网络; DNN 优化; DNN 缺陷; DNN 性能; 可解释性分析

中图法分类号 TP311.5

近年来, 深度神经网络(deep neural network, DNN)被成功应用于计算机视觉[1]、语音识别[2]、自然语言处理[3]等领域. 在自动驾驶[4]和智慧医疗[5]等场景下, DNN 甚至可以代替人类完成决策. 例如汽车制造商的自动驾驶技术[4]、监控系统常用的人脸识别技术[6]、工业制造使用的缺陷检测技术[7]等. 随着 DNN 应用领域的不断扩大, 人们对 DNN 的质量提出了更高要求. 尤其在一些安全攸关领域, 如果 DNN 的质量达不到要求, 或将引发灾难性后果. 据 NHTSA 统计[8], 在 2016—2021 年, 仅与特斯拉自动驾驶系统有关的 8 起交通事故中就导致了 10 人死亡. 因此, 如何有效保障 DNN 的质量已成为智能软件领域的一个重要研究课题.

与传统软件类似, DNN 也不可避免地含有缺陷. 对于传统软件的质量保障问题, 已有大量工作围绕缺陷检测[9]、缺陷定位[10]、增量调试[11]和自动错误修复[12]等技术展开研究, 并取得了良好的效果. 但上述质量保障技术无法直接适用于 DNN. 传统软件通常根据软件需求设计编写程序的内部逻辑, DNN 则是在定义好模型结构后使用数据集对模型进行训练, 使其学习输入数据到预测输出的高维映射关系. DNN 的内部行为逻辑取决于其结构、训练数据及训练得出的神经元权重等[13]. 因此, 需要根据 DNN 的特点研究有针对性的质量保障方法.

在 DNN 质量保障研究工作中, 通常认为准确率低于理想值就称为 DNN 存在缺陷[14]. 目前已有许多工作对 DNN 缺陷检测技术进行研究[15]. 例如 Pei 等人[16]根据 DNN 的结构特点提出了一种白盒差分测试技术, 在测试用例生成过程中将系统行为差异和高神经元覆盖率作为优化目标, 取得了较高的神经元覆盖率, 并在 Udacity 自动驾驶汽车挑战赛数据集中发现了撞击护栏等数千个错误行为. Zhang 等人[17]提出了基于对抗式生成网络(generative adversarial network, GAN)的测试数据生成方法 DeepRoad, 通过生成不同天气条件下的驾驶场景来检测基于 DNN 的自动驾驶系统的不一致行为.

在检测到 DNN 中存在的缺陷后, 更为重要的是针对所发现的缺陷优化 DNN, 以提升其性能. 现有的 DNN 优化工作主要通过扩增训练数据和优化 DNN 参数 2 种方式进行. 扩增训练数据是对训练数据进行扩增, 使用扩增后的数据训练 DNN 以提升性能. 例如 Ma 等人[14]提出了一种基于状态差分分析和输入选择的自动神经网络调试技术 MODE, 该方法可帮助识别有缺陷的神经元, 测量它们的重要性并使用 GAN 指导生成具有较高真实性的训练数据, 以提高模型性能. 优化 DNN 参数则直接调整 DNN 中的权重等参数以提升 DNN 性能. 例如, Zhang 等人[18]提出了一种 DNN 修复方法 Apricot, 其思想是使用部分原始数据集训练生成一组简化的 DNN, 然后利用它们提供搜索方向, 以调整待修复 DNN 的权重值. 上述 2 种方法均达到了较好的优化效果, 如 Apricot 方法在所有模型上的准确率平均提升了 1.08 个百分点, 提升的范围是 1.00~2.15 个百分点. MODE 方法的过程较为复杂且需要较高的训练成本, 而 Apricot 方法的修复过程需要构建训练多个 DNN 来提供搜索方向, 这将提高修复成本, 且修复过程不具有可解释性.

针对上述问题, 本文提出了一种基于可解释性分析的 DNN 优化方法(optimizing DNN based on interpretability analysis, OptDIA). 首先, 利用 DNN 解释方法计算不同区域对 DNN 预测结果的贡献值, 获取不同区域对 DNN 预测结果产生影响的重要程度. 然后, 对重要区域以不同概率使用遮挡、变异等方式进行数据变换以扩增训练数据集. 其中, 重要程度越低的区域进行数据变换的概率越高. 最后, 使用扩增后训练数据集再训练 DNN 以进行优化. 实验结果表明, OptDIA 可以将 DNN 的准确率提升 0.39~2.15 个百分点, *F*1-*score* 提升 0.11~2.03 个百分点.

本文的主要贡献有 2 个方面:

1) 提出了一种基于可解释性分析的 DNN 优化方法 OptDIA, 使用 DNN 解释方法指导扩增训练数据, 并通过再训练优化 DNN, 以达到提升其性能的目的.

2) 为验证 OptDIA 的有效性, 本文实现了原型工具, 对 9 个 DNN 模型在 3 个开源数据集上进行实验, 并对 OptDIA 优化模型性能的能力进行验证.

1 基于可解释性分析的 DNN 优化方法

1.1 动 机

目前，缺乏可解释性已经成为制约机器学习算法研究、开发和实现的一个关键问题. 随着 DNN 模型越来越多地用于自动驾驶、智慧医疗等安全攸关领域，人们对使用解释方法分析 DNN 的需求越来越大. 虽然已经提出了许多 DNN 解释方法，但使用解释方法分析 DNN 是否达到预期效果及提升 DNN 性能的研究相对较少. 由于训练数据会影响 DNN 性能，若使用解释方法对 DNN 预测过程进行分析，以理解其预测正确或错误的原因，可获取每条训练数据不同区域对 DNN 预测结果的影响；并进行有针对性的数据变换，可增加训练数据的多样性，提升模型性能. 为此，本文提出了基于可解释性分析的 DNN 优化方法 OptDIA.

OptDIA 的框架图如图 1 所示，主要包括 2 个部分. 首先使用解释方法对 DNN 的训练过程和决策行为进行分析以生成梯度加权类激活图[19]，梯度加权类激活图可显示原始数据中不同区域对 DNN 预测结果的重要程度. 若某块区域重要程度越高，则表示该区域所对应原始数据中的特征对预测结果影响越大(将在 1.2 节中详细介绍)；随后是基于可解释性分析的训练数据生成阶段，该阶段按一定概率选取训练数据，根据梯度加权类激活图进行数据变换操作生成新训练数据，对 DNN 再训练并记录训练过程中的性能，在达到停止条件时输出性能最佳的 DNN，并将其作为优化后的 DNN(将在 1.3 节中详细介绍).

1.2 DNN 解释分析及可视化

由于 DNN 具有复杂的多层非线性网络结构，难以全面理解其预测机制，因此常被当作黑盒使用，这也导致难以确认 DNN 产生缺陷的原因. 对 DNN 预测过程的内部工作机制进行深入分析，度量不同特征对预测结果的影响程度，才能有针对性地优化 DNN. 因此，OptDIA 中的一个关键步骤是对 DNN 的预测过程进行解释分析，以了解训练数据不同区域影响 DNN 行为的重要性差异.

目前，已有多种解释方法可用于理解 DNN 的内部工作机制，例如基于反向传播的 Grad[20]、基于特征反演的 Guide Inversion[21]、基于类激活映射的 CAM (class activation mapping)[22] 和基于梯度加权类激活映射的 Grad-CAM(gradient-weighted CAM)[19] 等. 相比于其他 DNN 解释方法，将梯度信息与特征映射相结合的 Grad-CAM 无需修改网络架构或重训练模型，可避免导致 DNN 准确率下降，并适用于不同任务以及多种结构的 DNN 模型. 因此，OptDIA 采用 Grad-CAM 方法对 DNN 进行解释分析和可视化.

OptDIA 首先会对训练数据进行解释分析. 具体地：在 DNN 的正向传播过程中，将最后一个卷积层作为 Grad-CAM 的目标层，并得到预测结果未经过 Softmax 激活之前的预测值，再通过反向传播计算目标层所输出的各个特征图对于预测值的梯度，将梯度全局平均池化后得到特征图对于预测结果类别的贡献值. 目标层特征图对于预测值的贡献值计算方法如式(1)[19]：

$$\alpha_k^c = \frac{1}{Z}\sum_i\sum_j \frac{\partial y^c}{\partial A_{ij}^k}. \tag{1}$$

其中，c 表示类别，y^c 是在经过 Softmax 激活之前 DNN 对类别 c 的预测值，A^k 表示目标层的第 k 个特征图，i, j 分别为特征图 A^k 的单个像素值在横轴和纵轴的索引，Z 为特征图 A^k 的长和宽的乘积. α_k^c 即第 k 个特征图对于类别 c 的贡献值.

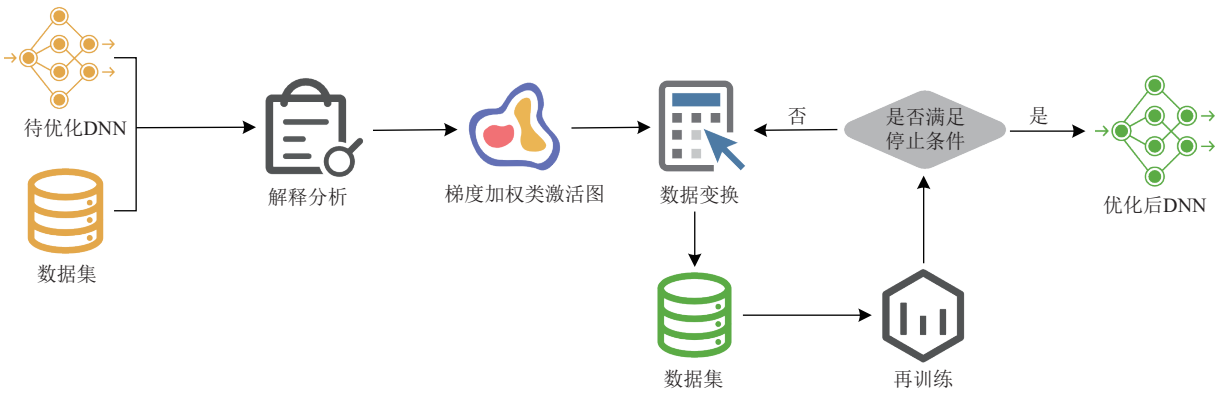


Fig. 1 The framework of OptDIA

图 1 基于可解释性分析的 DNN 优化方法框架

接下来, 将特征图对于预测类别的贡献值作为权重, 对目标层的所有特征图进行加权求和, 并通过函数 *ReLU* 将所有负值取 0, 消除一些与目标类别无关的干扰, 从而获得一个粗粒度的梯度加权类激活图, 以用于定位每条训练数据中具有类判别性的重要区域. 其中, 类判别性指的是训练数据中对 DNN 预测提供重要依据的特征. 类别 c 的梯度加权类激活图计算方法如式(2)所示[19]:

$$L^{c}_{\text{Grad-CAM}} = ReLU\left(\sum_{k} a^{c}_{k} A^{k}\right). \tag{2}$$

在获取梯度加权类激活图后, 可将其与原始数据叠加生成热力图, 以更加直观地了解不同区域对 DNN 预测结果的重要性差异. 图 2 为解释方法 Grad-CAM 生成的梯度加权类激活图, 以及与原始数据叠加生成的热力图示例. 图 2 中每一行的图片分别为来自 CIFAR-10, CIFAR-100[23], Fashion-MNIST[24] 数据集中的训练数据, 标签分别为“bird”“man”“t-shirt”. 图 2 中第 1 列为原始数据; 第 2 列为使用 Grad-CAM 所生成的梯度加权类激活图; 第 3 列为将梯度加权类激活图与原始数据叠加生成的热力图, 其中颜色越深的区域对 DNN 分类结果影响越大. 从图 2 可以看出; 对于图片“bird”, 对 DNN 分类结果起重要作用的部分为其纺锤形的体型特征; 对于图片“man”, 对 DNN 分类结果起重要作用的部分为其身体特征; 对于图片“t-shirt”, 对 DNN 分类结果起重要作用的部分为衣服袖口.

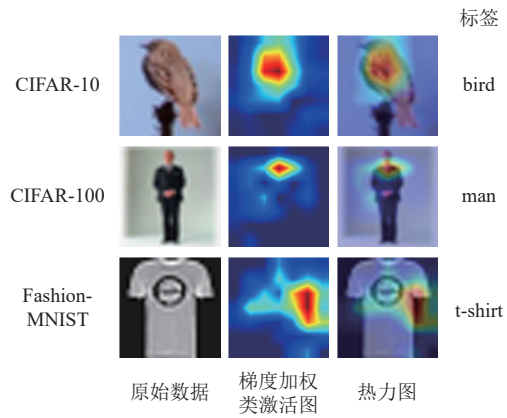


Fig. 2 Example of gradient-weighted class activation map and heat map on different datasets

图 2 不同数据集的梯度加权类激活图和热力图示例

1.3 基于可解释性分析的训练数据生成

DNN 性能受模型结构、超参数和训练数据等因素影响, 在模型结构和超参数确定的情况下, 训练数据的质量和多样性等因素也会影响 DNN 性能. 因此如何提高训练数据的质量及多样性是 OptDIA 的关键. 目前, 几何变换[25]、颜色空间增强[26]、核滤波器[27]、随机擦除[28]、对抗训练[29] 和神经风格转移[30] 等数据增强方法常用于优化训练数据. 但是这些增强训练数据方法通常被研究者用来解决小样本训练问题. OptDIA 将基于解释分析结果应用数据增强方法提高训练数据质量, 以优化 DNN, 进而提升其性能.

基于可解释性分析的训练数据生成步骤如算法 1 所示.

算法 1. 基于可解释性分析的训练数据生成.

输入: 训练集 *trainSet*, 测试集 *testSet*, 原始数据索引 *imgId*, 待优化 DNN *M*;

输出: 优化后 DNN *M′*.

① for each *i* in *imgId*

② *camMap* = *GradCAM*(*M*, *trainSet*[*i*]);

③ *areaRank* = *getImportantArea*(*camMap*);

④ *trainSet*[*i*] = *Augment*(*areaRank*, *trainSet*[*i*]);

⑤ end for

⑥ *models* = *reTrain*(*trainSet*, *testSet*, *M*);

⑦ *bestAcc* = 0;

⑧ for each *m* in *models*

⑨ *acc* = *validate*(*testSet*, *m*);

⑩ if *acc* > *bestAcc*

⑪ *bestAcc* = *acc*;

⑫ *M′* = *m*;

⑬ end if

⑭ end for

⑮ return *M′*.

算法 1 中, 首先将训练数据输入到 DNN 进行解释分析, 以生成梯度加权类激活图 *camMap*(行①②). 然后, 根据 *camMap* 统计固定大小区域的热力值以获取原始数据中对预测结果影响重要性程度区域排序 *araRank*(行③), 其中, 热力值指训练数据中不同区域对 DNN 预测结果的重要性程度, 值越大越重要. 接着, 根据原始数据中的重要性程度区域排序 *areaRank* 使用数据增强方法 *Augment* 对原始数据中的一个区域进行数据变换(其中重要程度排序越低的区域被选择进行数据变换的概率越高), 并将所生成的数据替换训练数据集中的原始数据(行④), 其中变换数据方法可以采用随机擦除[28] 等数据增强技术. 训练数据替换完成后, 对 DNN 进行重训练并验证每个训练轮次中 DNN 的准确率(行⑧⑨). 最后, 在达到预设定的重训练轮数后停止优化, 并从中选择准确率最高的 DNN *M′*作为优化后的 DNN 输出(行⑩~⑮).

此算法的最坏时间复杂度为 $O(n)$，其中 n 为数据集规模.

基于算法 1，通过使用数据增强方法对梯度加权类激活图中的不同区域按重要程度和不同概率选择进行变换生成新的训练数据，以提高训练数据的质量及多样性，并使用生成的数据再次训练 DNN 以提升其性能完成优化. 图 3 为 OptDIA 方法生成新数据示例. 其中，第 1 列为对来自 CIFAR-10, CIFAR-100, Fashion-MNIST 数据集中的原始数据使用 Grad-CAM 所生成的梯度加权类激活图，标签分别为“bird”“man”“t-shirt”. 第 2 列为根据梯度加权类激活图生成的新数据，用于再训练. 梯度加权类激活图用作变换数据范围的依据以生成新数据，将对 DNN 预测起重要作用的区域以较小的概率变换，而其他对 DNN 预测结果不重要的区域以较大的概率进行数据变换，来让 DNN 学习更多重要区域的特征. 例如对于“bird”，让 DNN 学习更多身体部位特征；对于“man”，让 DNN 学习更多人体形态特征；对于“t-shirt”，让 DNN 学习更多衣服特征. OptDIA 通过解释分析对预测结果不重要的特征进行变换，来让 DNN 学习更多的重要特征，从而正确分类测试数据.

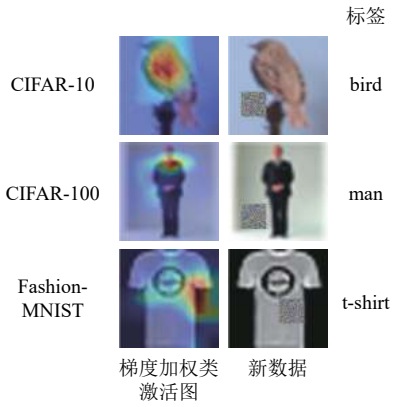


Fig. 3 Example of training data generated based on interpretability analysis on different datasets

图 3 不同数据集基于可解释性分析的训练数据生成示例

2 实验设计及评估

2.1 实验设计

我们基于所提出的 OptDIA 方法，实现了 DNN 优化原型工具. 在原型工具中，通过解释分析对原始数据中的区域进行重要性排序，排序越高的区域被选择使用随机擦除方法[28]进行数据变换的概率越低. 其中，随机擦除指的是对被选择进行数据变换的区域用随机值像素替换以生成具有不同遮挡程度的训练图像，每条数据被选择擦除的概率沿用文献 [28] 中的 0.1. 为确定实验所用的擦除区域大小，分别将图片区域的分割方式设置为 64, 16, 4 个大小相等的矩形区域，初步实验结果表明，不同区域大小对 OptDIA 的优化效果并无显著影响. 在实验中我们选择将图像等分为 16 个大小相等的矩形区域，CIFAR-10 和 CIFAR-100 数据集的矩形大小为 8×8, Fashion-MNIST 数据集的矩形大小为 7×7. 原型工具的开发和运行环境为 Python3.6 和 Pytorch1.12.

本文通过 3 个研究问题来验证 OptDIA 的有效性.

问题 1. OptDIA 是否能优化 DNN 以提升 DNN 性能.

OptDIA 是一种基于可解释性分析的 DNN 优化方法，为了评估 OptDIA 的有效性，设计了问题 1 验证 OptDIA 在不同数据集训练的不同 DNN 模型上提升 DNN 性能的有效性.

问题 2. 解释方法的引入是否提高了 OptDIA 优化 DNN 的有效性.

解释方法的引入是 OptDIA 区别于其他 DNN 优化方法的重要步骤，为了评估解释方法的引入对 DNN 优化能力的影响，设计了问题 2 验证 OptDIA 在 DNN 优化过程中，使用解释方法选择数据变换区域和使用随机方法选择数据变换区域对优化效果的影响.

问题 3. OptDIA 能否在已使用数据增强方法进行训练的模型上继续提升 DNN 性能.

目前，数据增强方法已经能有效地在训练过程中对训练数据进行扩增，以提高模型的泛化能力并达到较高的性能. 问题 3 验证 OptDIA 是否能进一步提升应用了数据增强方法进行训练的 DNN 性能.

2.2 实验环境

2.2.1 数据集

为回答 2.1 节的 3 个问题，实验将 OptDIA 在使用 CIFAR-10, CIFAR-100, Fashion-MNIST 数据集训练的 DNN 上进行验证和分析. 所使用的数据集被广泛应用于提升 DNN 性能的工作中[14,18,28]. 其中，CIFAR-10 数据集由 10 类共 60 000 张尺寸为 32×32 的 3 通道彩色图像组成，其中包含 50 000 张训练图像和 10 000 张测试图像. CIFAR-100 具有和 CIFAR-10 相同的数据格式和规模，2 个数据集的区别在于类别数不同，CIFAR-10 具有 10 种分类类别，而 CIFAR-100 具有 100 种分类类别. Fashion-MNIST 由 10 类共 70 000 张尺寸为 28×28 的单通道灰度图像组成，其中包含 60 000 张训练图像和 10 000 张测试图像. 数据集的详细信

Table 1 The Statistics of Datasets

表 1 数据集信息

数据集	总图像数	总类别数	训练集图像数	测试集图像数
CIFAR-10	60 000	10	50 000	10 000
CIFAR-100	60 000	100	50 000	10 000
Fashion-MNIST	70 000	10	60 000	10 000

息如表 1 所示. 数据集已经被划分好训练集及测试集, 实验使用其中的训练集训练, 测试集用来测试及评估优化效果.

2.2.2 实验对象

我们使用 CIFAR-10, CIFAR-100, Fashion-MNIST 数据集训练了 3 种模型: 分别使用 18 层、34 层和 50 层结构的 ResNet[31]. 优化器学习率从 0.01 开始每训练 20 轮下降 3 倍, 在第 50 轮停止训练. 将这些完成训练的 DNN 作为待优化模型. 为减少随机性的影响, 所有实验均重复运行 3 次后取平均值.

2.2.3 评价标准

在实验中使用准确率 *Accuracy* 和 *F*1-*score* 来评价优化效果, 以反映 DNN 的拟合能力和预测未知样本的能力.

对于二分类问题, 其中正样本被正确预测称为 *TP*, 负样本被错误预测称为 *FP*, 负样本被正确预测称为 *TN*, 正样本被错误预测称为 *FN*. 式(3)为准确率 *Accuracy* 的计算方式, 即所有样本中正确预测为正样本占所有样本的比例; 式(4)为精确率 *Precision* 的定义, 即预测的正样本中实际是正样本的个数占被预测为正样本的比例; 式(5)为召回率 *Recall* 的定义, 即所有样本中被预测为正样本占所有正样本的比例; 式(6)为 *F*1-*score* 的定义, 同时考虑精确率和召回率. 实验中使用的 CIFAR-10, CIFAR-100, Fashion-MNIST 数据分别有 10, 100, 10 个类别. 对于多分类问题, 可将每个数据集中的类两两组合, 分别计算每个类和其他类之间的 *Accuracy* 和 *F*1-*score*, 再计算平均值即可.

$$Accuracy=\frac{TP+TN}{TP+TN+FN+FP}, \tag{3}$$

$$Precision=\frac{TP}{TP+FP}, \tag{4}$$

$$Recall=\frac{TP}{TP+FN}, \tag{5}$$

$$F1\text{-}score=\frac{2\times Precision\times Recall}{Precision+Recall}. \tag{6}$$

2.3 实验结果及分析

2.3.1 针对问题 1 的结果分析与讨论

为了评估 OptDIA 的有效性, 在实验中使用固定大小的滑动窗口扫描原始数据, 对每个矩形区域统计其在加权梯度类激活图中的贡献值, 并根据贡献值排序, 贡献值越低的区域进行数据变换的概率越高. 实验将分别在 3 个数据集上预训练 50 轮的 9 个 DNN 模型作为待优化模型, 再使用 OptDIA 重训练 50 轮进行优化, 并将预训练 300 轮的 DNN 模型的准确率与优化后 DNN 模型的准确率进行对比.

实验结果如表 2 所示, 其中 Org_{50} 和 Org_{300} 分别表示优化前预训练 50 轮和 300 轮 DNN 的准确率, OptDIA 表示优化后 DNN 的准确率. 实验结果表明, OptDIA 可提升使用 3 个数据集训练的 9 个 DNN 的准确率以进行优化. 其中, 对于使用 CIFAR-10 训练的 ResNet-50 模型, OptDIA 将准确率提升了 1.23 个百分点; 对于使用 CIFAR-100 训练的 ResNet-50 模型, OptDIA 将准确率提升了 2.15 个百分点; 对于使用 Fashion-MNIST 训练的 ResNet-50 模型, OptDIA 将准确率提升了 1.32 个百分点. 实验结果表明, 在 3 个数据集上训练的 9 个模型中, 有 7 个使用 OptDIA 优化 50 轮后所提升的准确率比直接预训练 300 轮模型的准确率高. 其中, 对于使用 CIFAR-10 训练的 ResNet-50 模型, 优化 50 轮后的准确率要比预训练 300 轮模型准确率高 1.02 个百分点; 对于使用 CIFAR-100 训练的 ResNet-50 模

Table 2 Accuracy Comparison of Different DNNs Before and After Optimization by OptDIA

表 2 不同 DNN 使用 OptDIA 优化前后的准确率比较 %

模型	CIFAR-10			CIFAR-100			Fashion-MNIST		
	Org_{50}	Org_{300}	OptDIA	Org_{50}	Org_{300}	OptDIA	Org_{50}	Org_{300}	OptDIA
ResNet-18	85.56	85.83	86.09	49.95	49.78	50.50	92.60	93.10	93.49
ResNet-34	84.59	84.51	85.39	55.22	59.13	56.61	92.98	93.56	93.51
ResNet-50	83.96	84.17	85.19	55.66	56.02	57.81	92.18	92.31	93.48

型，优化 50 轮后的准确率要比预训练 300 轮模型准确率高 1.79 个百分点；对于使用 Fashion-MNIST 训练的 ResNet-18 模型，优化 50 轮后的准确率要比预训练 300 轮模型准确率高 0.39 个百分点. 其中，对于初始准确率较低的模型，准确率提升空间比较大，OptDIA 提升准确率的幅度也相对较大. 如使用 CIFAR-100 训练的 ResNet-50 模型，初始准确率 55.66%，OptDIA 将其准确率提升了 2.15 个百分点；对于初始准确率较高的模型，其准确率提升空间较小，OptDIA 提升准确率的幅度也相对较小. 如使用 Fashion-MNIST 训练的 ResNet-34 模型，初始准确率已达 92.98%，OptDIA 仅将其准确率提升了 0.53 个百分点.

从表 2 可以看出，对于 3 种 DNN 模型，使用 OptDIA 均可提升其准确率，甚至相比较预训练 300 轮的 DNN 模型，预训练 50 轮后使用 OptDIA 优化的 DNN 模型准确率更高. 对实验结果进行分析发现，这是因为 OptDIA 使用解释方法对 DNN 的预测过程进行分析，获取每条训练数据中不同区域特征对 DNN 结果预测的重要程度，并对越不重要的区域以越大的概率进行变换生成新数据，增加训练数据的多样性，从而提升 DNN 模型的性能.

对问题 1 的回答：对于使用不同数据集和结构训练的 DNN，OptDIA 均可有效提升其准确率，提升幅度为 0.39~2.15 个百分点，所有模型的准确率平均提升了 1.04 个百分点，即使与训练 300 轮的 DNN 模型相比，使用 OptDIA 优化的模型准确率也更高.

2.3.2 针对问题 2 的结果分析与讨论

为分析 OptDIA 中解释方法对 DNN 模型优化效果的影响，实验在不同结构的 DNN 和数据集上分别使用 OptDIA-和 OptDIA 优化方法，并对所提升的性能进行比较. 其中，OptDIA-指没有使用 Grad-CAM 对数据进行解释分析，而是直接对训练数据进行随机擦除操作以扩增训练数据. 实验在预训练 50 轮的待优化 DNN 基础上，分别用 OptDIA-和 OptDIA 再训练 50 轮，并对比所提升性能的差异.

表 3 给出了 OptDIA-和 OptDIA 使用 3 个数据集训练的 9 个 DNN 模型上提升性能的情况. 如表 3 所示，除 ResNet-50 模型在 CIFAR-100 数据集上使用 OptDIA-的优化效果优于 OptDIA 外，其他模型使用 OptDIA 的优化效果均优于比 OptDIA-，准确率提升了 0.39~2.15 个百分点，*F*1-*score* 提升了 0.11~2.03 个百分点. 其中，对于使用 CIFAR-10 训练的 ResNet-18，OptDIA 可以提升 0.53 个百分点的准确率和 0.41 个百分点的 *F*1-*score*，而 OptDIA-只能提升 0.20 个百分点的准确率和 0.17 个百分点的 *F*1-*score*，OptDIA 所提升的准确率和 *F*1-*score* 分别为 OptDIA-的 2.65 倍和 2.41 倍；对于使用 CIFAR-10 训练的 ResNet-34，OptDIA 可以提升 0.80 个百分点的准确率和 0.87 个百分点的 *F*1-*score*，而 OptDIA-只能提升 0.15 个百分点的准确率和 0.78 个百分点的 *F*1-*score*，OptDIA 所提升的准确率和 *F*1-*score* 分别为 OptDIA-的 5.33 倍和 1.12 倍.

就所有模型的平均优化效果而言，对使用 CIFAR-10 数据集训练的 DNN，OptDIA 和 OptDIA-平均提升的准确率为 0.85 个百分点和 0.54 个百分点，提升的 *F*1-*score* 为 0.90 个百分点和 0.85 个百分点，OptDIA 平均提升的准确率和 *F*1-*score* 分别是 OptDIA-的 1.57 倍和 1.06 倍；对使用 CIFAR-100 数据集训练的 DNN，OptDIA 和 OptDIA-平均提升的准确率分别为 1.36 个百分点和 1.21 个百分点，提升的 *F*1-*score* 分别为 0.90 个百分点和 1.24 个百分点，OptDIA 平均提升的准确率是 OptDIA-的 1.12 倍，平均提升的 *F*1-*score* 与 OptDIA-

Table 3 Performance Improvements of Optimizing Different DNNs by OptDIA- and OptDIA

表 3 不同 DNN 使用 OptDIA-和 OptDIA 优化性能提升比较

模型	评价指标	CIFAR-10		CIFAR-100		Fashion-MNIST	
		OptDIA-/%	OptDIA/%	OptDIA-/%	OptDIA/%	OptDIA-/%	OptDIA/%
ResNet-18	*Accuracy*	0.20	0.53	0.33	0.55	0.53	0.60
	*F*1-*score*	0.17	0.41	0.02	0.24	0.11	0.69
ResNet-34	*Accuracy*	0.15	0.80	1.12	1.39	0.32	0.39
	*F*1-*score*	0.78	0.87	0.72	0.16	0.53	0.11
ResNet-50	*Accuracy*	1.26	1.23	2.36	2.15	1.21	1.32
	*F*1-*score*	0.16	0.14	2.99	2.03	1.23	1.5
平均值	*Accuracy*	0.54	0.85	1.21	1.36	0.69	0.77
	*F*1-*score*	0.85	0.90	1.24	0.90	0.82	1.24

相近; 对使用 Fashion-MNIST 数据集训练的 DNN, OptDIA 和 OptDIA-平均提升的准确率为 0.77 个百分点和 0.69 个百分点, 提升的 *F*1-*score* 为 1.24 个百分点和 0.82 个百分点, OptDIA 平均提升的准确率和 *F*1-*score* 分别是 OptDIA-的 1.12 倍和 1.47 倍.

从表 3 可以看出, 使用解释方法指导变换数据再训练 DNN 所提升的性能高于未使用解释方法指导变换数据再训练 DNN 所提升的性能. 对实验结果进行分析发现, 这是因为解释方法可更好地帮助理解 DNN 模型的预测过程. 使用解释方法将 DNN 训练过程透明化, 以掌握训练数据中部分区域特征的重要程度, 并将此作为变换数据以增加其多样性的依据. 而未使用解释方法指导数据变换则有较强随机性, 难以使 DNN 模型学习多样化的特征.

OptDIA 使用解释方法分析 DNN 的预测行为, 并根据分析结果对原始数据进行数据变换以生成新数据. 解释方法的引入将消耗更高的时间成本, 过高的额外时间成本将影响 OptDIA 在现实场景中的实用性. 为研究 OptDIA 在有效提高 DNN 准确率的前提下额外消耗的时间成本, 在计算表 3 中的实验数据时, 我们还统计了对不同结构的 DNN 和数据集分别使用 OptDIA-和 OptDIA 生成新数据所消耗的时间成本, 如表 4 所示.

如表 4 所示, 使用解释方法 Grad-CAM 选择擦除区域的 OptDIA, 与直接对训练数据进行随机擦除的 OptDIA-相比, 解释方法的引入平均额外消耗了 1 min 34 s, 即平均每生成一张新训练数据额外消耗约 0.016 s (以擦除概率为 0.1, 每个数据集生成约 6 000 条新数据计算). OptDIA 只对被选择进行数据变换的数据进行解释分析, 算法的时间复杂度为 $O(n)$, 且分析和数据变换区域的选择均可由 GPU 执行, 具有较高的执行效率. 与 DNN 动辄数小时的训练耗时相比, OptDIA 额外消耗的时间成本并不会明显降低其实用性.

对问题 2 的回答: 与随机选择数据变换区域的方法相比, OptDIA 能更有效优化 DNN 以提升其性能, 且消耗额外时间较少.

2.3.3 针对问题 3 的结果分析与讨论

为验证 OptDIA 是否能在应用现有数据增强方法进行训练的 DNN 上进行优化, 首先将 Zhong 等人[28]提出的数据增强方法, 即随机擦除(random erasing, RE), 应用于 DNN 训练, 然后再对使用 RE 训练的 DNN 尝试使用 OptDIA 进行优化. 实验使用 RE 方法在 3 个数据集上分别训练了 300 轮的 9 个 DNN 模型, 并在其基础上再使用 OptDIA 继续训练 50 轮.

实验结果如表 5 所示, 其中 RE 表示不同结构的 DNN 使用随机擦除方法进行数据增强训练所得到 DNN 的准确率, OptDIA 表示在使用 RE 方法训练的基础上再使用 OptDIA 继续训练所得到 DNN 的准确率. 实验结果表明, 与直接训练相比, RE 训练的模型准确率提升了 0.51~3.24 个百分点, OptDIA 则进一步提升使用 RE 训练的 6 个模型的准确率和 7 个模型的 *F*1-*score*. 其中, 对使用 CIFAR-10 数据集训练的模型, OptDIA 将 RE 训练的 ResNet-50 模型的准确率和 *F*1-*score* 分别进一步提升了 0.32 个百分点和 0.97 个百分点, 对使用 CIFAR-100 数据集训练的模型, OptDIA 将 RE 训练模型的 ResNet-50 模型的准确率和 *F*1-*score* 进一步提升了 2.72 个百分点和 2.35 个百分点, 对使用 Fashion-MNIST 数据集训练的模型, 其准确率和 *F*1-*score* 已较高, OptDIA 将 RE 训练的 ResNet-50 模

Table 4 Time Costs Comparison of Different DNNs by Using OptDIA- and OptDIA to Generate Training Data

表 4 不同 DNN 使用 OptDIA-和 OptDIA 生成训练数据的时间消耗比较

模型	数据集	OptDIA-	OptDIA
ResNet-18	CIFAR-10	1 min 05 s	2 min 03 s
	CIFAR-100	1 min 05 s	1 min 59 s
	Fashion-MNIST	1 min 01 s	2 min 05 s
ResNet-34	CIFAR-10	1 min 07 s	2 min 36 s
	CIFAR-100	1 min 06 s	2 min 34 s
	Fashion-MNIST	1 min 01 s	2 min 41 s
ResNet-50	CIFAR-10	1 min 07 s	3 min 09 s
	CIFAR-100	1 min 06 s	3 min 08 s
	Fashion-MNIST	1 min 03 s	3 min 30 s
平均值		1 min 04 s	2 min 38 s

Table 5 Performance Comparison of Different DNNs Trained by Using RE and Before and After Optimization by OptDIA

表 5 不同 DNN 使用 RE 和 OptDIA 优化前后性能比较

模型	评价指标	CIFAR-10		CIFAR-100		Fashion-MNIST	
		RE/%	OptDIA（本文）/%	RE/%	OptDIA（本文）/%	RE/%	OptDIA（本文）/%
ResNet-18	*Accuracy*	87.29	87.57	53.02	53.64	93.95	93.95
	*F*1-*score*	87.28	87.41	53.34	53.73	93.94	93.94
ResNet-34	*Accuracy*	87.69	87.80	57.30	58.41	94.07	94.07
	*F*1-*score*	87.66	87.94	57.46	58.12	94.05	93.89
ResNet-50	*Accuracy*	85.83	86.15	56.02	58.81	93.59	93.84
	*F*1-*score*	85.84	86.31	56.25	58.60	93.59	93.74

型的准确率和 *F*1-*score* 分别进一步提升了 0.25 个百分点和 0.15 个百分点.

从表 5 可以看出, 对于经使用数据增强方法预训练的 DNN 模型, OptDIA 仍可进一步提升其准确率, 即达到优化 DNN 的效果. 对实验结果进行分析发现, 因为利用 DNN 的解释方法指导进行数据变换能有效增加训练样本的多样性, 使 DNN 学到更多的特征.

对问题 3 的回答: OptDIA 可进一步提升使用数据增强方法训练的 DNN 模型的准确率和 *F*1-*score*, 提升幅度分别为 0.11~2.72 个百分点和 0.13~2.35 个百分点.

2.4 有效性分析

2.4.1 内部有效性

本文方法的内部有效性分析主要在于确实影响实验正确性的因素. 首先, 实验中使用的 DNN 解释方法为 Grad-CAM 的开源项目 pytorch-grad-cam[32], 该开源项目也被其他相关工作使用[33]; 其次, 我们参考开源项目 Random-Erasing 的源代码[34] 实现了实验所用的数据增强方法; 最后, 实验构建和训练 DNN 模型的代码来自 Torchvision[35] 的开源实现, 此外, 还对代码进行了多次内部交叉检查, 以尽量确保实现的正确性.

2.4.2 外部有效性

基于可解释性分析的深度神经网络优化方法的外部有效性主要在于实验中所用的 DNN 模型和数据集是否具有代表性以及 OptDIA 是否具有可扩展性. 首先, 本文实验所使用的 ResNet 模型以及 CIFAR-10, CIFAR-100, Fashion-MNIST 数据集在优化 DNN 的工作中被广泛使用[14,18,28], 具有一定代表性, 但不能确保 OptDIA 在其他待优化 DNN 和数据集上的有效性. 其次, 在实验中只使用了擦除方法对数据进行扩增, 不能确保 OptDIA 能在裁剪等其他数据扩增方法上的有效性. 此外, 虽然任何准确率的提升都是对模型有意义的优化[14], 但由于 DNN 模型具有内在不确定性, 因此无法保证准确率的提升幅度.

2.4.3 构造有效性

本文方法的构造有效性影响主要在于优化 DNN 的评价指标. 实验中使用测试集的准确率和 *F*1-*score* 来评价实验效果, 这 2 项指标常被用于评价 DNN 性能[14]. 此外, 为避免实验结果的随机性, 我们使用相同参数进行多次实验求得平均值作为实验结果.

3 相关工作

传统软件缺陷修复方法通过基于启发式搜索[9]、人工模板[36]、语义约束[37] 和统计分析[38] 等技术自动生成针对特定缺陷的程序补丁, 并将补丁自动添加到软件以修复缺陷, 或将补丁作为提示信息来帮助开发者继续优化软件[12]. 与传统软件不同, OptDIA 主要关注基于 DNN 的智能软件, 其内部具体权重参数通过训练获取且结构较为复杂, 这使其输出结果缺乏传统软件的解释性, 因此传统方法难以直接应用于优化 DNN. 目前对 DNN 进行优化的研究主要从训练集和模型参数 2 个切入点进行, 本节将对这 2 类方法分别进行介绍.

一类方法是通过扩增训练集优化 DNN 以提升其性能. Ma 等人[14] 提出了 DNN 调试方法 MODE, 该方法首先通过分析神经网络状态生成差分热图 (differential heat map), 并结合 DNN 模型找出对于分类影响较大的神经元, 分别对在训练过程中存在欠拟合或过拟合的 DNN 进行调试, 然后指导 GAN 生成具有较高真实性的训练数据, 从而提升模型的性能以优化 DNN. Gao 等人[39] 提出了一种基于模糊测试的数据扩充方法, 该方法结合模糊测试和遗传算法扩增数据, 将优化问题转化为搜索问题, 以提高模型的健壮性, 从而达到优化 DNN 的目的. Liu 等人[40] 提出了一个面向数据的变异框架 Styx, 该框架首先

通过对原始数据进行轻微变异生成新的训练集, 并将其用于重新训练 DNN, 以提高模型健壮性, 从而达到优化的目的. 文献 [14, 39−40] 所述的方法可在一定程度上优化 DNN, 然而依然存在扩充数据集的过程缺乏针对性, 需要引入新的 DNN, 增加数据多样性的能力有限且过程难以理解等问题.

另一类方法是通过修改模型参数优化 DNN 以提升其性能. Zhang 等人[18] 提出了一种权重调整方法 Apricot, 该方法通过在原始训练集的许多不同子集上进行训练得到简化深度学习模型(reduced deep learning model, rDLM), 使用其权重来辅助修改原始模型中的权重, 从而提升模型的准确率以优化 DNN. 随后他们又提出一种超启发式方法 Plum[41], 该方法通过应用不同的修复策略生成一组候选 DNN 模型, 然后根据候选修复策略的整体优化效果对修复策略进行评估和排序, 并使用最佳优化策略生成优化后的 DNN. 基于变异的 DNN 优化方法 GenMuNN[42] 量化神经元权重对 DNN 预测结果的影响, 并根据量化结果对各层神经元权重进行排序, 然后使用遗传算法对权重进行变异以生成变异体, 直到出现准确率较高的 DNN, 即可达到优化的目的. Zhang 等人[43] 提出了一种 DNN 训练监控与自动修复方法 AUTOTRAINER, 该方法可检测梯度消失、梯度爆炸、损失振荡和收敛缓慢等常见训练问题, 并自动采用如替换激活函数等内置修复方案提高 DNN 性能, 以达到优化 DNN 的目的. Sun 等人[44] 提出了基于因果关系的神经网络修复方法 CARE, 该方法通过最小限度地调整给定 DNN 的权重参数, 通过执行基于因果关系的缺陷定位以识别并优化“有罪(guilty)”神经元, 通过构建一个满足目标属性(如公平性、无后门等)的神经网络来提高其性能. 然而, 直接对 DNN 参数进行优化也会有一定风险, 例如在调整权重时, 直接调整参数难以控制修改幅度, 甚至可能会影响模型准确地拟合训练数据.

文献 [18, 41−44] 所述的工作主要通过扩增数据集或修改模型参数的方式优化 DNN, 由于缺乏解释性导致其修复过程大多难以理解. 本文所提出的基于可解释性分析的深度神经网络优化方法 OptDIA 使用 DNN 解释方法获取训练数据中对预测结果影响较大的区域, 并通过对其进行数据变换生成新的数据再训练 DNN, 以提升 DNN 性能, 从而达到优化 DNN 的目的. 相较之前的工作, OptDIA 引入了 DNN 解释方法, 提高了数据扩增的针对性, 使优化 DNN 的过程更容易被理解.

4 总结和未来工作

为优化 DNN 以提升其性能, 本文提出了一种基于可解释性分析的 DNN 优化方法 OptDIA. 使用解释方法获取原始数据中对 DNN 决策具有较高重要性的区域, 并将其作为变换数据生成新数据的依据, 再重新训练 DNN 以提升其性能, 从而达到优化 DNN 的目的. 在使用 3 个数据集训练的 9 个 DNN 模型上进行的实验结果表明, OptDIA 可以有效提升 DNN 性能以进行优化, 并可进一步优化使用现有数据增强方法训练的 DNN. 在下一步工作中, 我们计划在更大规模数据集上引入更多数据增强方法验证 OptDIA 的优化效果, 并计划尝试将本文方法引入语音、文本处理等领域, 如利用解释方法获取词语对情感分析结果的重要性程度, 将其作为变换数据的依据, 使用变换后的数据重新训练模型, 以提高其性能.

作者贡献声明: 吴欢欢负责相关工作调研、论文撰写及修订等工作; 谢瑞麟完成实验并修改论文; 乔塬心协助完成实验; 陈翔提出指导意见并修改论文; 崔展齐指导论文选题、确定论文整体框架设计并修改论文.

参 考 文 献

[1] Esteva A, Chou K, Yeung S, et al. Deep learning-enabled medical computer vision[J]. NPJ Digital Medicine, 2021, 4(1): 1−9

[2] Malik M, Malik M K, Mehmood K, et al. Automatic speech recognition: A survey[J]. Multimedia Tools and Applications, 2021, 80(6): 9411−9457

[3] Zheng Haibin, Chen Jinyin, Zhang Yan, et al. Survey of adversarial attack, defense and robustness analysis for natural language processing[J]. Journal of Computer Research and Development, 2021, 58(8): 1727−1750 (in Chinese)
(郑海斌, 陈晋音, 章燕, 等. 面向自然语言处理的对抗攻防与鲁棒性分析综述 [J]. 计算机研究与发展, 2021, 58(8): 1727−1750)

[4] Fang Peng, Wang Zecong, Zhang Xiangyun. Vehicle automatic driving system based on embedded and machine learning[C]//Proc of Int Conf on Computer Vision, Image and Deep Learning. Piscataway, NJ: IEEE, 2020: 281−284

[5] Bakator M, Radosav D. Deep learning and medical diagnosis: A review of literature[J]. Multimodal Technologies and Interaction, 2018, 2(3): 47

[6] Ghenescu V, Mihaescu R E, Carata S V, et al. Face detection and recognition based on general purpose DNN object detector[C]//Proc

of Int Symp on Electronics and Telecommunications. Piscataway, NJ: IEEE, 2018: 1−4

[7] Tulbure A A, Tulbure A A, Dulf E H. A review on modern defect detection models using DCNNs-Deep convolutional neural networks[J]. Journal of Advanced Research, 2022, 35: 33−48

[8] Interface News. Review of new energy vehicles[EB/OL]. 2022[2023-02-16].https://www.jiemian.com/article/7530592.html (in Chinese)
(界面新闻. 新能源汽车评论[EB/OL]. 2022[2023-02-16].https://www.jiemian.com/article/7530592.html)

[9] Goues C L, Nguyen T V, Forrest S, et al. GenProg: A generic method for automatic software repair[J]. IEEE Transactions on Software Engineering, 2011, 38(1): 54−72

[10] Baudry B, Fleurey F, Traon Y L. Improving test suites for efficient fault localization[C]//Proc of the 28th Int Conf on Software Engineering. New York: ACM, 2006: 82−91

[11] Zeller A. Yesterday, my program worked. Today, it does not. Why?[J]. ACM SIGSOFT Software Engineering Notes, 1999, 24(6): 253−267

[12] Zhang Xiangyu, Gupta N, Gupta R. Locating faults through automated predicate switching[C]//Proc of the 28th Int Conf on Software Engineering. New York: ACM, 2006: 272−281

[13] Ma Lei, Xu Felix Juefei, Zhang Fuyuan, et al. Deepgauge: Multi-granularity testing criteria for deep learning systems[C]//Proc of the 33rd ACM/IEEE Int Conf on Automated Software Engineering. New York: ACM, 2018: 120−131

[14] Ma Shiqing, Liu Yingqi, Lee W C, et al. MODE: Automated neural network model debugging via state differential analysis and input selection[C]//Proc of the 26th ACM Joint Meeting on European Software Engineering Conf and Symp on the Foundations of Software Engineering. New York: ACM, 2018: 175−186

[15] Wang Zan, Yan Ming, Liu Shuang, et al. A survey on testing of deep neural network. [J] Journal of Software, 2020, 31(5): 1255−1275 (in Chinese)
(王赞, 闫明, 刘爽, 等. 深度神经网络测试研究综述[J]. 软件学报, 2020, 31(5): 1255−1275)

[16] Pei Kexin, Cao Yinzhi, Yang Junfen, et al. Deepxplore: Automated whitebox testing of deep learning systems[C]//Proc of the 26th Symp on Operating Systems Principles. New York: ACM, 2017: 1−18

[17] Zhang Mengshi, Zhang Yuqun, Zhang Lingming, et al. DeepRoad: GAN-based metamorphic testing and input validation framework for autonomous driving systems[C]//Proc of the 33rd IEEE/ACM Int Conf on Automated Software Engineering. Piscataway, NJ: IEEE, 2018: 132−142

[18] Zhang Hao, Chan W K. Apricot: A weight-adaptation approach to fixing deep learning models[C]//Proc of the 34th IEEE/ACM Int Conf on Automated Software Engineering. Piscataway, NJ: IEEE, 2019: 376−387

[19] Selvaraju R R, Cogswell M, Das A, et al. Grad-CAM: Visual explanations from deep networks via gradient-based localization [C]//Proc of the IEEE Int Conf on Computer Vision. Piscataway, NJ: IEEE, 2017: 618−626

[20] Simonyan K, Vedaldi A, Zisserman A. Deep inside convolutional networks: Visualising image classification models and saliency maps[J]. arXiv preprint, arXiv: 1312. 6034, 2013

[21] Du Mengnan, Liu Ninghao, Song Qinghao, et al. Towards explanation of DNN-based prediction with guided feature inversion[C]//Proc of the 24th ACM SIGKDD Int Conf on Knowledge Discovery & Data Mining. New York: ACM, 2018: 1358−1367

[22] Zhou Bolei, Khosla A, Lapedriza A, et al. Learning deep features for discriminative localization[C]//Proc of the IEEE Conf on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2016: 2921−2929

[23] CIFAR official website. Dataset[EB/OL]. [2023-02-16]. http://www.cs.toronto.edu/~kriz/cifar.html

[24] GitHub. Fashion-MNIST[EB/OL]. 2017[2023-02-16].https://github.com/zalandoresearch/fashion-mnist

[25] Mikołajczyk A, Grochowski M. Data augmentation for improving deep learning in image classification problem[C]//Proc of 2018 Int Interdisciplinary PhD Workshop (IIPhDW). Piscataway, NJ: IEEE, 2018: 117−122

[26] Taylor L, Nitschke G. Improving deep learning with generic data augmentation[C]// Proc of IEEE Symp Series on Computational Intelligence. Piscataway, NJ: IEEE, 2018: 1542−1547

[27] Kang Guoliang, Dong Xuanyi, Zheng Liang, et al. Patchshuffle regularization[J]. arXiv preprint, arXiv: 1707. 07103, 2017

[28] Zhong Zhun, Zheng Liang, Kang Guoliang, et al. Random erasing data augmentation[C]//Proc of the AAAI Conf on Artificial Intelligence. Palo Alto, CA: AAAI, 2020: 13001−13008

[29] Zajac M, Zołna K, Rostamzadeh N, et al. Adversarial framing for image and video classification[C]//Proc of the AAAI Conf on Artificial Intelligence. Palo Alto, CA: AAAI, 2019: 10077−10078

[30] Johnson J, Alahi A, Li Feifei. Perceptual losses for real-time style transfer and super-resolution[C]//Proc of the 14th European Conf on Computer Vision. Berlin: Springer, 2016: 694−711

[31] He Kaiming, Zhang Xiangyu, Ren Shaoqing, et al. Deep residual learning for image recognition[C]//Proc of the IEEE Conf on Computer Vision and Pattern Recognition. Piscataway, NJ: IEEE, 2016: 770−778

[32] GitHub. Grad-CAM[EB/OL]. 2022[2023-02-16].https://github.com/jacobgil/pytorch-grad-cam

[33] Chen Lei, Chen Jianhui, Hajimirsadeghi H, et al. Adapting Grad-CAM for embedding networks[C]//Proc of the IEEE/CVF Winter Conf on Applications of Computer Vision. Piscataway, NJ: IEEE, 2020: 2794−2803

[34] GitHub. Random-Erasing[EB/OL]. 2019[2023-02-16].https://github.com/zhunzhong07/Random-Erasing

[35] GitHub. Torchvision[EB/OL]. 2022[2023-02-16].https://github.com/pytorch/vision

[36] Tian Yuchi, Ray B. Automatically diagnosing and repairing error handling bugs in C[C]//Proc of the 11th Joint Meeting on Foundations of Software Engineering. New York: ACM, 2017: 752−762

[37] Nguyen H D T, Qi Dawei, Roychoudhury A, et al. Semfix: Program repair via semantic analysis[C]//Proc of the 35th Int Conf on Software Engineering. Piscataway, NJ: IEEE, 2013: 772−781

[38] Jeffrey D, Feng Min, Gupta N, et al. BugFix: A learning-based tool to assist developers in fixing bugs[C]//Proc of the 17th IEEE Int Conf on Program Comprehension. Piscataway, NJ: IEEE, 2009: 70−79

[39] Gao Xiang, Saha R K, Prasad M R, et al. Fuzz testing based data augmentation to improve robustness of deep neural networks [C]//Proc of the 42nd IEEE/ACM Int Conf on Software Engineering. Piscataway, NJ: IEEE, 2020: 1147−1158

[40] Liu Meixi, Hong Weijiang, Pan Weiyu, et al. Styx: A data-oriented mutation framework to improve the robustness of DNN[C]//Proc of the 35th IEEE/ACM Int Conf on Automated Software Engineering. Piscataway, NJ: IEEE, 2020: 1260−1261

[41] Zhang Hao, Chan W K. Plum: Exploration and prioritization of model repair strategies for fixing deep learning models[C]//Proc of the 8th Int Conf on Dependable Systems and Their Applications. Piscataway, NJ: IEEE, 2021: 140−151

[42] Wu Huanhuan, Li Zheng, Cui Zhanqi, et al. GenMuNN: A mutation-based approach to repair deep neural network models[J]. International Journal of Modeling, Simulation, and Scientific Computing, 2022, 13(2): 2341008

[43] Zhang Xiaoyu, Zhai Juan, Ma Shiqing, et al. AUTOTRAINER: An automatic DNN training problem detection and repair system[C]//Proc of the 43rd IEEE/ACM Int Conf on Software Engineering. Piscataway, NJ: IEEE, 2021: 359−371

[44] Sun Bing, Sun Jun, Pham L H, et al. Causality-based neural network repair[C]//Proc of the 44th Int Conf on Software Engineering. New York: ACM, 2022: 338−349



Wu Huanhuan, born in 1995. Master. Student member of CCF. Her main research interests include intelligent software engineering and trustworthy artificial intelligence.

吴欢欢, 1995 年生. 硕士. CCF 学生会员. 主要研究方向为智能软件工程、可信人工智能.



Xie Ruilin, born in 1996. Master candidate. Student member of CCF. His main research interests include intelligent software engineering and trustworthy artificial intelligence.

谢瑞麟, 1996 年生. 硕士研究生. CCF 学生会员. 主要研究方向为智能软件工程、可信人工智能.



Qiao Yuanxin, born in 1999. Master candidate. Student member of CCF. His main research interests include intelligent software engineering and trustworthy artificial intelligence.

乔塬心, 1999 年生. 硕士研究生. CCF 学生会员. 主要研究方向为智能软件工程、可信人工智能.



Chen Xiang, born in 1980. PhD, associate professor, master supervisor. Senior member of CCF. His main research interests include software testing, software maintenance, empirical software engineering, and mining software repository.

陈　翔, 1980 年生. 博士, 副教授, 硕士生导师. CCF 高级会员. 主要研究方向为软件测试、软件维护、经验软件工程、软件仓库挖掘.



Cui Zhanqi, born in 1984. PhD, professor, master supervisor. Senior member of CCF. His main research interests include intelligent software engineering and trustworthy artificial intelligence.

崔展齐, 1984 年生. 博士, 教授, 硕士生导师. CCF 高级会员. 主要研究方向为智能软件工程、可信人工智能.