

面向移动应用自动化测试的同构用户界面视觉判断方法

薛峰¹, 武君胜², 张涛², 王威², 成静³

(1.西北工业大学 计算机学院, 陕西 西安 710072; 2.西北工业大学 软件学院, 陕西 西安 710072; 3.西安工业大学 计算机科学与工程学院, 陕西 西安 710021)

摘要:当前快速增长的移动应用程序迫切需求自动化测试技术以保证其质量。移动应用的自动化测试与其图形用户界面(GUI)的识别与判断紧密相关,但移动应用却通常存在大量的样式与内容有差异而结构与功能相类似的同构GUI。在自动化测试中,同构GUI容易引发应用状态空间的爆炸问题,进而导致测试的低效或失败。针对传统自动化识别同构GUI的局限性,提出一种基于视觉特征信息的GUI相似度判断方法。通过目标检测技术识别GUI组件元素进而抽象出GUI结构框架;采用卷积自编码器提取出GUI结构视觉特征;对比GUI视觉特征的相似度完成同构GUI判断。经过实验验证,所提方法能够屏蔽GUI的样式、内容等影响,从而更精确地完成同构GUI识别,优化自动化测试效率。

关键词:移动应用测试;GUI视觉特征;同构GUI;GUI相似判断

中图分类号:TP311 **文献标志码:**A **文章编号:**1000-2758(2022)04-0804-08

随着移动应用的日益普及和快速发展,移动应用测试正面临严峻的挑战^[1-4]。首先,移动应用存在基于Android、iOS、Windows Mobile以及Mobile Web等平台的多样化开发模式,但却缺乏支持多平台应用的通用化测试方法与测试工具。其次,移动设备品牌、型号众多,操作系统版本繁杂,其面临严重的碎片化问题,并很难达成有效且全面的测试。并且,移动设备提供了更加丰富的人机交互方式,例如,手势、语音,以及重力、加速度、光等多种传感器,使得其输入复杂,加剧移动应用测试难度。最后,移动市场竞争日益激烈,移动应用版本更新频繁,开发商普遍缺乏足够的时间、人员、设备、工具等测试资源。

为应对上述测试难题,学术界与工业界持续投入对移动应用自动化测试方法的研究。当前的自动化测试方法主要包括:基于脚本的测试方法通过构建自动化测试框架将人工编写的测试脚本转换为模

拟的事件流实现对测试动作在移动应用上的模拟执行^[5-6];基于录制回放的测试方法将人工测试的过程进行录制进而通过回放重用该执行过程^[7-8];基于模型的自动化测试方法通过对被测应用进行定义和遍历将应用的功能行为构建成为一种有限状态机模型实现测试用例的自动生成^[9-10];另有通过挖掘已有测试用例并对比应用功能的相似性实现测试用例的重用^[11-12]。此外,随着机器人技术的发展,一些采用机械臂执行移动应用测试的方法被提出^[13-14],即利用机械臂替代传统模拟事件流的测试执行,提供一种更真实的测试手段;同时,解决移动应用与外部交互的测试问题,如利用机械臂对拍照应用进行防抖动测试^[15]。基于机器人的测试可作为一种完全的黑盒测试方法支持跨平台、跨设备的应用测试。

从上述各类测试方法研究中可以看出移动应用测试通常紧密围绕于GUI状态的判断,进而引导测

收稿日期:2021-09-15

作者简介:薛峰(1987—),西北工业大学博士研究生,主要从事软件工程、软件测试研究。

通信作者:张涛(1976—),西北工业大学副教授,主要从事软件工程、软件系统架构研究。e-mail:tao_zhang@nwpu.edu.cn

试动作交互完成测试。然而,当前的自动化测试却始终面临一个关键性的问题即移动应用状态空间的爆炸^[16]。在移动应用中通常存在大量的同构 GUI,即 GUI 的外观(文本、图像、颜色、大小)不同,但功能、结构以及内在逻辑关系相同的 GUI。例如,购物类应用每个商品的展示页面,其 GUI 结构相同但内容不同。在开发层面,它们均通过同一代码段加载实现。显然,对移动应用中的每一同构 GUI 进行探索或测试是十分低效的。甚至,当应用存在大量的同构 GUI 时,将最终导致应用状态空间的爆炸,引发测试的失败。

因此,在自动化测试方法中,同构 GUI 的识别与精简对提升测试效率至关重要。而当前测试方法在同构 GUI 的判断上,一方面通过分析 GUI 布局文件获取由 GUI 组件构成的 GUI 树,从而比较相似度;另一方面直接通过对 GUI 或 GUI 元素图像进行像素比较完成相似判断。但是采用上述方式仍然使得 GUI 的判断与 GUI 属性相关联,即使在 GUI 发生微小的变化时也会引发判断的失效。例如,一个列表中内容的重新排序,或者在版本更新后 GUI 元素的位置或样式发生略微变动均会影响判断。然而,在人工测试中,因为可以进行充分抽象化的 GUI 比较,所以并不受限于这些影响。

针对上述问题,本文提出一种基于 GUI 视觉信息的相似度抽象判断方法,即从视觉角度出发通过提取 GUI 结构框架,保留 GUI 元素类别、布局等关键信息,剥离 GUI 元素样式、内容等非必要属性信息,以实现一种从布局结构出发的高度抽象化的移动应用同构 GUI 判断。

1 同构 GUI 视觉判断方法

移动应用同构 GUI 视觉判断方法的处理过程如图 1 所示。当从被测应用获取到当前的 GUI 图像时,首先通过 GUI 视觉抽象处理识别 GUI 图像中的组件元素并生成 GUI 结构框架,完成对 GUI 结构的提取;其次,利用一个自编码器模型得到 GUI 框架的视觉特征;最后,通过与 GUI 特征库中已知被测应用的 GUI 特征进行相似度计算判断该 GUI 是否为同构 GUI。倘若其为非同构 GUI(即一个新的 GUI)则将该特征加入特征库中并继续执行对该 GUI 的测试。倘若该 GUI 为同构 GUI,则跳过该 GUI 的测试(说明之前已测试同类型 GUI),优化测试过程。本节将给出移动应用同构 GUI 视觉判断方法各处理部分的详细实现。

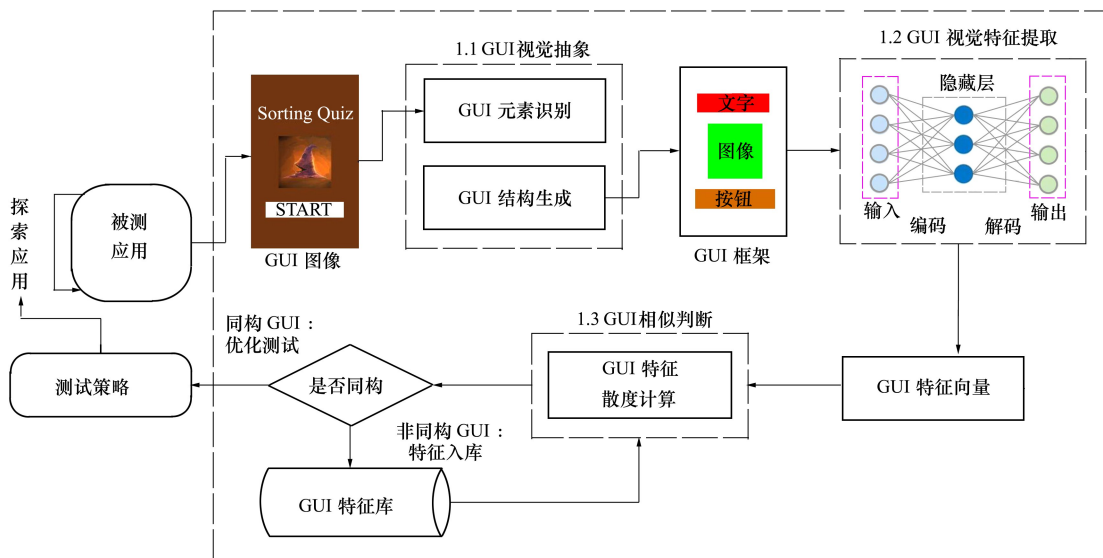


图 1 移动应用同构 GUI 视觉判断处理过程

1.1 移动应用 GUI 的视觉抽象

移动应用 GUI 通常由丰富的组件元素构成,因此在移动应用 GUI 的视觉判断研究中,首先解决对

GUI 组件元素的视觉定位与识别。GUI 界面从布局结构上通常为一个树形结构,由容器类组件包含基本组件所构成。然而,从视觉角度出发,使用者更关

注于界面上直观的基本元素以及其所能执行的操作。因此在 GUI 元素的视觉识别上将聚焦于对基本组件元素的识别。此外,在一个 GUI 图像中,容器类组件一般自身并未有明显的视觉特征,而是由基本组件所组合而成。因此,识别基本组件并未丢失容器类组件的视觉表示信息。

本文采用目标检测技术完成对 GUI 组件元素的视觉识别。首先,在移动应用 GUI 目标检测数据集的准备方面,基于已有的大型 GUI 界面数据集 RICO^[17] 开展,其具有涵盖多个类别的 6 万幅移动应用 GUI 截屏,且每个截屏配备一个 GUI 层次结构文件(包含 GUI 中各组件所属类及位置信息)。在充分考虑组件特性和视觉特征的前提下,将识别组件类别分为文本、图像、图形按钮、文字按钮、文本框、多选框、复选框、切换按钮、拖动条。进而利用 RICO 提供的 GUI 层次结构文件,编写转换程序,完成其向目标检测所需标注文件的过滤与转换,通过自动化的标注方式完成移动应用 GUI 目标检测数据集的建立。

其次,开展目标检测算法对 GUI 组件元素的识别训练。为进一步降低单个模型检测出现的漏检等问题并优化识别精度,本文基于移动应用 GUI 目标检测数据集分别训练 3 种目标检测模型(YOLO^[18]、SSD^[19]、RetinaNet^[20]),并采用并行投票策略完成 GUI 组件元素识别。

最后,依据 GUI 组件元素识别结果生成 GUI 结构框架图,抽象表示出 GUI 结构。GUI 框架图常用于 GUI 的设计与开发中^[21-22],是一种对 GUI 结构的表示方法。本文针对 GUI 组件元素类型,以不同的颜色标识其类别,生成 GUI 框架图。将 GUI 表示为 GUI 框架图的目的在于:①剥离掉 GUI 元素样式、大小、形状的差异性,提取 GUI 的结构表示信息;②可以简化图像对比的计算复杂度。GUI 结构框架生成过程如图 2 所示。

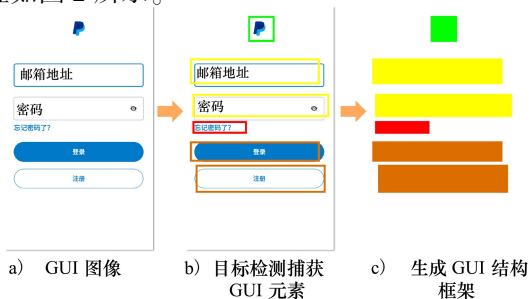


图 2 移动应用 GUI 结构框架生成

1.2 移动应用 GUI 结构视觉特征提取

在生成 GUI 结构框架后,利用构建自编码器模型继续提取 GUI 结构视觉特征。自编码器(AutoEncoder)是一种经典的无监督学习方法,其面向高维复杂数据处理,通过使用反向传播算法让目标值等于输入值。例如,自编码器可以通过编码器生成图像的特征向量,再通过解码器依据特征向量生成目标图像。通过比较原图像与生成图像的差异,无监督地完成图像特征向量的准确生成。

传统的自编码器容易造成数据在固定空间内的堆叠,导致关键信息的丢失,因此本文采用一种卷积自编码器^[23]尽可能保留图像特征。即在自编码器的结构上利用卷积层替代全连接层从而更好地完成图像特征的提取。

本文定义的卷积自编码器结构如图 3 所示,包括编码器和解码器两部分。编码器由卷积层和池化层组成,负责完成对输入 GUI 框架图的压缩。经过 4 组卷积与池化层后的隐藏层即为可提取的 GUI 框架图特征向量。解码器与编码过程相对,通过 4 组卷积层和上采样层,对压缩数据进行复原重建输入图像。自编码器经过反复编码与解码的迭代训练,逐步提高自编码精度,直至解码后的数据和输入数据的差距逐渐减少并趋于平稳。当完成自编码器的训练,编码器则可以提取任意输入 GUI 框架图视觉特征。下面给出本文卷积自编码器的详细设计。

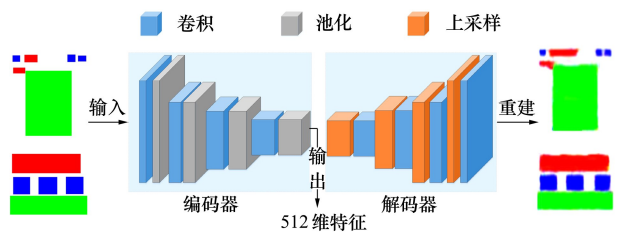


图 3 基于卷积自编码器的 GUI 视觉特征提取

1) 数据预处理

GUI 框架图在输入到卷积自编码器之前,首先进行归一化处理以提高自编码器模型训练性能,加速收敛。采用 Min-Max 标准的归一化方法,把原始数据从 [0, 255] 归一化到标准的 [0, 1] 范围内,如 (1) 式所示。

$$n_i = \frac{m_i - m_{\min}}{m_i - m_{\max}} \quad (1)$$

式中: n_i 为规范化的第 i 个数据,值为 [0, 1] 之间; m_i 为未规划前的数据; m_{\min} 为原数据中的最小值; m_{\max}

为原数据中的最大值。

2) 编码器设计

编码器包括卷积层和池化层,对输入数据进行特征提取和压缩。编码器由4个卷积和4个池化构成。卷积层的功能用于提取和融合输入图像特征,它不识别整个图像而是通过局部感知来捕获图像中的每个特征,然后在更高层次上对本地信息进行综合运算得到全局信息。利用生成的GUI框架图作为输入数据,其维度为 $128 \times 128 \times 3$ 。编码器4个卷积,依次使用64,32,16,8个大小为 3×3 的卷积核,步长默认为1,填充设置为相同。此外,选择ReLU(校正线性单元层)作为激活函数,进行卷积层线性运算到GUI结构视觉特征输出的非线性映射。

在卷积层提取到GUI框架特征后,输出特征将被传送到编码器的池化层进行特征选择和信息过滤。池化层用于对特征图进行下采样处理,在保留重要信息的前提下快速降低特征维度以减少计算量,并避免过拟合提高模型的容错性。经过第一层卷积后,数据维度由原来的 $128 \times 128 \times 3$ 变为 $128 \times 128 \times 64$ 。池化操作不改变数据的深度,所以池化后的数据维度为 $64 \times 64 \times 64$ 。池化层对每层卷积结果进行 2×2 最大池化操作,步长为2。经过4层卷积和池化操作,得到隐藏层的压缩表示。

3) 解码器设计

编码器对GUI结构视觉特征进行提取和压缩后,进一步使用解码器对特征进行反卷积操作以获得与输入数据相同的大小。解码器由4个上采样和4个卷积构成。采用的反卷积操作由第一步上采样和第二步正常的卷积操作组成,即通过上采样将特征图像扩大,再进行卷积操作而完成反卷积过程。上采样采用最近邻插值方法进行,即寻找插入位置的最邻近数据作为其数据插入。解码器中上采样每层使用大小为 2×2 ,步长为2的操作,卷积层依次使用8,16,32,64个大小为 3×3 的卷积核,步长设置为1,填充设置为相同,与编码器的卷积层相对应。

4) 损失层

损失层用于确定在训练过程中如何减少网络预测结果和实际结果之间的差异。损失函数值越小,模型拟合越好。在本文的卷积自编码器中,采用均方根误差作为损失函数计算输入GUI框架原图与重建框架图之间的差异。损失函数如(2)式所示。

$$L_{\text{MSE}} = \frac{1}{n} \sum_i^n (x_i - f(x_i))^2 \quad (2)$$

式中: n 为输入的图片数量; x_i 为输入第 i 张图片; $f(x_i)$ 为重建的第 i 张图片。在训练过程中,GUI框架原图像和重建图像之间的差异将不断减小,直到模型收敛。完成训练后,原图像的压缩编码数据即为GUI结构的视觉特征。

1.3 移动应用GUI相似判断

提取到GUI框架的视觉特征后,进一步通过计算GUI框架特征的距离度量判断GUI相似情况。本文采用JS散度进行GUI特征的距离度量。

JS(Jensen-Shannon)散度是机器学习中经常用来衡量随机分布之间相似度的方法。任意2个GUI随机分布使用以2为底的对数,JS散度范围为 $[0, 1]$,即GUI相似度范围在 $[0, 1]$ 之间。JS散度越小GUI越相似,即JS散度值为0时,2个GUI完全一样,JS散度值为1时,2个GUI没有重叠的地方。JS散度定义为(3)式。

$$J(\mathbf{G}_1, \mathbf{G}_2) = \frac{1}{2} \sum_{i=1}^n g_{1i} \log_2 \frac{g_{1i}}{\frac{g_{1i}}{2} + \frac{g_{2i}}{2}} + \frac{1}{2} \sum_{i=1}^n g_{2i} \log_2 \frac{g_{2i}}{\frac{g_{1i}}{2} + \frac{g_{2i}}{2}} \quad (3)$$

式中: $J(\mathbf{G}_1, \mathbf{G}_2)$ 为2个GUI相似度量值; \mathbf{G}_1 和 \mathbf{G}_2 为2个GUI特征集合; n 为GUI分布数量即每个GUI特征向量个数,本文取 $n = 512$; g_{1i} 为 \mathbf{G}_1 的GUI集合中第 i 个随机值; g_{2i} 为 \mathbf{G}_2 的GUI集合中第 i 个随机值。

GUI相似判断算法如算法1所示。 \mathbf{g}_1 作为已知GUI结构特征, \mathbf{g}_2 为当前探索到某一GUI特征。利用JS散度计算出各维度的相似情况,并通过阈值判断是否相似。最后,返回与 \mathbf{g}_2 相似的GUI信息(相似GUI,相似度,是否同构)。

在计算JS散度时,某一特征数据为0时为避免计算错误,将其设置为一个极小值 10^{-7} 。此外,因JS散度越小GUI越相似,为直观体现相似性结果,用1减去散度作为相似判断结果。

算法1 GUI相似判断

输入:已知GUI特征向量集合 \mathbf{G} ,当下的GUI特征向量 \mathbf{g}_2

输出:相似GUI信息 $P < \text{GUI}$,相似度,是否同构 $>$

```

1   $J_{\text{sm}} = 0$ 
2  for each  $\mathbf{g}_1$  in  $\mathbf{G}$  do
3       $j \leftarrow \text{getJSDivergence}(\mathbf{g}_1, \mathbf{g}_2)$ 

```

```

4   if  $j > \text{threshold}$  then
5        $P \leftarrow \text{getJudgeInfo}(g_1, j, 1)$ 
6       return  $P$ 
7   else if  $j > J_{sm}$  then
8        $J_{sm} = j$ 
9        $P \leftarrow \text{getJudgeInfo}(g_1, j, 0)$ 
10  end if
11 end for
12 return  $P$ 

```

至此,通过视觉技术生成 GUI 结构框架,利用自编码器提取 GUI 框架特征,再比较 GUI 框架特征相似度完成同构 GUI 的视觉判断。

2 实验验证分析

为验证本文提出的移动应用同构 GUI 视觉判断方法的有效性,分别开展 GUI 视觉特征提取的准确性验证,同构 GUI 判断的有效性验证以及同构 GUI 判断对比验证。

2.1 移动应用 GUI 视觉特征提取验证

为验证 GUI 视觉特征提取的准确性,从 RICO 数据集中选取 1 000 张 GUI 并生成其 GUI 框架图作为特征提取数据集,其中,800 张作为训练集,200 张作为测试集。卷积自编码器输入 $128 \times 128 \times 3$ 结构的图像数据,首先进行归一化处理,其次经过编码器的卷积、最大池化和解码器中的上采样、卷积处理,最后再经过一次卷积后恢复为输入的图片尺寸。使用 Adam 优化算法作为网络优化参数,优化训练。批处理大小 (batch_size) 设置为 32,迭代轮次 (epochs) 设置为 2 000。训练环境采用 NVIDIA GeForce GTX 2080 GPU,基于 pycharm 训练平台的 keras 深度学习框架。在模型训练结束后,输入测试集进行 GUI 框架视觉特征提取验证。最后提取的每张图像压缩后的数据大小为 $8 \times 8 \times 8 = 512$ 个特征。卷积自编码器模型结构如表 1 所示。

模型的训练结果如图 4 所示。模型的损失函数,在训练集和测试集上分别收敛于 0.02 和 0.03,模型精度分别为 0.83 和 0.81。

GUI 框架原图和由卷积自编码器重建框架图像对比如图 5 所示,重建图像与原图像越相似说明 GUI 特征提取的越准确。从图 5 中可以直观地看到,重建图像与原图具有较高的相似度,因此本文的卷积自编码器能够准确地提取 GUI 框架视觉特征。

表 1 卷积自编码器模型结构

| 层级 | 输出维度 | 参数规模 |
|----------------|----------------|--------|
| Conv2D | (128, 128, 64) | 1 792 |
| MaxPooling2D | (64, 64, 64) | 0 |
| Conv2D_1 | (64, 64, 32) | 18 464 |
| MaxPooling2D_1 | (32, 32, 32) | 0 |
| Conv2D_2 | (32, 32, 16) | 4 624 |
| MaxPooling2D_2 | (16, 16, 16) | 0 |
| Conv2D_3 | (16, 16, 8) | 1 160 |
| MaxPooling2D_3 | (8, 8, 8) | 0 |
| UpSampling2D | (16, 16, 8) | 0 |
| Conv2D_4 | (16, 16, 8) | 584 |
| UpSampling2D_1 | (32, 32, 8) | 0 |
| Conv2D_5 | (32, 32, 16) | 1 168 |
| UpSampling2D_2 | (64, 64, 16) | 0 |
| Conv2D_6 | (64, 64, 32) | 4 640 |
| UpSampling2D_3 | (128, 128, 32) | 0 |
| Conv2D_7 | (128, 128, 64) | 18 496 |
| Conv2D_8 | (128, 128, 3) | 1 731 |

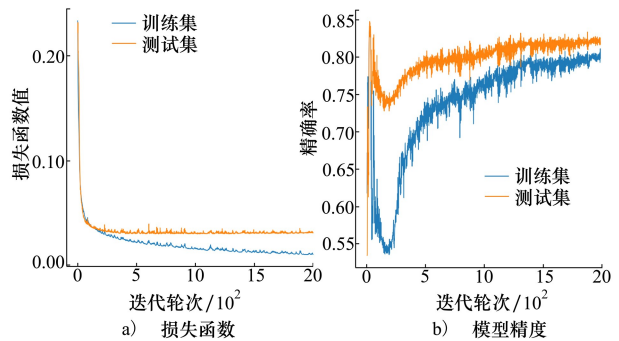


图 4 模型训练结果

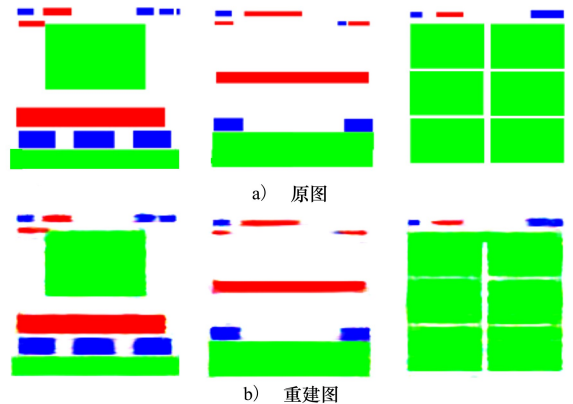


图 5 GUI 框架原图与重建图对比

2.2 有效性验证

为进一步验证同构 GUI 判断的有效性,额外选取包含同构 GUI 的 100 张 GUI 进行实验。其中,前 50 张作为已知 GUI,后 50 张为需要判断的 GUI。图

6a)所示为真实值,对角线表示对应的图像为同构 GUI,其他区域为非同构 GUI。对角线上网格的大小代表该同构界面的数量,同构界面数量越多网格越大。利用文本方法将后 50 张 GUI 分别与前 50 张 GUI 进行共计 2 500 次判断计算。

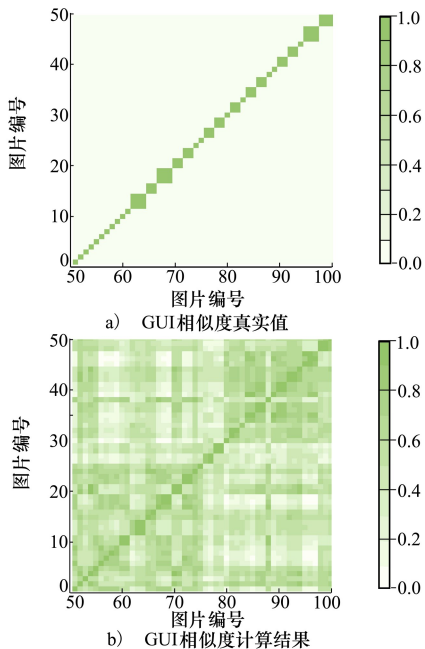


图 6 GUI 相似度判断验证结果

图 6b)所示为前后 2 组 GUI 之间的相似度计算结果,设置相似度阈值为 0.9,大于为同构 GUI,小于为非同构 GUI。图中颜色越深,表示相似度值越高。对比图 6a)中同构 GUI 的真实值,在图 6b)中可以明显看出大于 0.9 的同构 GUI 和真实值 GUI 所在位置基本一致,而小于 0.9 的非同构 GUI 所在位置总体为浅色和白色所覆盖。对角线同构界面的颜色明显,且网格大小与真实值相近。因此,实验表明本文方法能够依据视觉信息正确判断同构 GUI。

然而,结果仍然存在一些误差的原因主要在于:一方面,在方法层面,视觉判断方法依赖于目标检测等视觉处理模型精度,例如 GUI 组件的类别错误检测或漏检均会影响相似度计算结果;另一方面,在 GUI 的特性层面,GUI 具备丰富的视觉特征和灵活的设计风格,应用界面内容的多样化会影响同构判断。

2.3 对比验证

另外,分别与传统的 3 种图像哈希算法进行对比实验。哈希算法为图像分配唯一的哈希值,并通过对比图像的哈希值判断相似度,结果越接近则图

像越相似。选择差值哈希算法、感知哈希算法、均值哈希算法 3 种基于哈希的图像相似度比较算法进行对比验证。通过构建 GUI 灰度图像并计算比较哈希值完成。

利用准确率、精确率、召回率、F1 值进行结果的评估分析。各方法表现如图 7 所示。

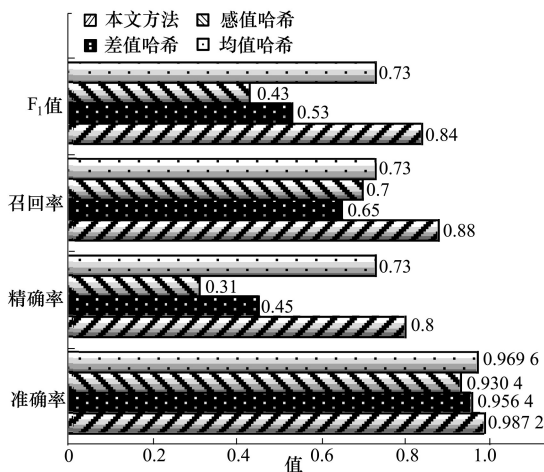


图 7 对比验证结果

对比实验结果表明,尽管 4 种方法的准确率均大于 0.9,但本文方法达到最高的准确率 0.987 2。而在精确率、召回率、F1 值评价指标上,本文方法均达到 0.8 以上,其效果远优于其他方法。本文因采用对 GUI 的高度抽象进而再相似比较,相较于其他方法有效屏蔽了 GUI 样式、内容等属性所带来的干扰影响,能够更为精准地捕捉同构 GUI。

2.4 讨论

上述实验验证表明,本文提出的基于视觉的移动应用同构识别方法已经达到了较高的识别精度。利用视觉判断同构 GUI 与传统非视觉的判断方法(例如,GUI 树比较判断)相比较存在如下优劣:①视觉判断方法通过应用 GUI 外在视觉表示信息进行判断,而非视觉方法则依赖于应用内部组件组成信息。因此,视觉判断方法可以作为一种支持跨平台或黑盒测试的通用方法,而非视觉判断方法则更针对单一平台应用或白盒测试;②视觉判断方法是针对 GUI 抽象结构的判断,而非视觉判断方法则是精确的 GUI 组件及属性比较。尽管视觉判断方法受到目标检测等视觉算法精度的影响,但却可以屏蔽 GUI 组件属性微小改动而带来的判断失效影响;③视觉判断方法拥有更高的使用成本。视觉判断方法需要进行多种视觉算法的计算与训练,需要更高

的计算与时间投入。综上分析,本文提出的视觉判断方法是对现有同构 GUI 判断的补充与增强,促进其在跨系统平台,设备非侵入以及纯黑盒机器人测试等场景下的使用。

3 结 论

本文提出了一种基于视觉的移动应用同构 GUI 视觉判断方法,通过利用目标检测技术以纯视觉方式获取 GUI 组件元素从而生成 GUI 结构框架,并引入卷积自编码器完成对 GUI 结构视觉特征的提取,最后通过散度计算比较 GUI 结构视觉特征来判断 GUI 相似度,形成一种高度抽象化的 GUI 判断分析。实验表明本文方法能够准确地识别同构 GUI,促进

解决移动应用自动化测试所面临的状态空间爆炸问题。此外,该方法因为不依赖于移动应用内部信息,可灵活应用于现有的自动化白盒与黑盒测试中。

本文方法仍然存在一些可改进之处。首先,部分应用存在灵活的界面表达方式,进一步复杂化同构界面的自动化判断。除了 GUI 结构上的比较,加入语义信息将会增强对此类多样化界面的判断。其次,采用视觉方法识别 GUI 存在一定的精度误差,将视觉判断方法与非视觉判断方法相融合将促进同构界面判断效果的增强。后续,仍可持续深入移动应用 GUI 在同表示含义,同行为模式上的视觉特征判断方法研究,进一步强化移动应用测试中的 GUI 自动化分析与理解。

参考文献:

- [1] KONG P, LI L, GAO J, et al. Automated testing of Android apps: a systematic literature review[J]. IEEE Trans on Reliability, 2018, 68(1): 45-66
- [2] TRAMONTANA P, AMALFITANO D, AMATUCCI N, et al. Automated functional testing of mobile applications: a systematic mapping study[J]. Software Quality Journal, 2019, 27(1): 149-201
- [3] LINARES-VÁSQUEZ M, BERNAL-CÁRDENAS C, MORAN K, et al. How do developers test android applications? [C]// 2017 IEEE International Conference on Software Maintenance and Evolution, 2017: 613-622
- [4] RUBINOV K, BARESI L. What are we missing when testing our android apps? [J]. Computer, 2018, 51(4): 60-68
- [5] Appium-automation for apps[EB/OL].(2018-10-05)[2021-08-18]. <http://appium.io/dols/cn/about-appium/intro>
- [6] Eyeautomate-visual script runner[EB/OL].(2019-02-08)[2021-08-12]. <https://eyeautomate.com/eyeautomate>
- [7] AMALFITANO D, RICCIO V, AMATUCCI N, et al. Combining automated GUI exploration of android apps with capture and replay through machine learning[J]. Information and Software Technology, 2019, 105: 95-116
- [8] GUO J, LI S, LOU J G, et al. SARA: self-replay augmented record and replay for android in industrial cases[C]//Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis, 2019: 90-100
- [9] GU T, SUN C, MA X, et al. Practical GUI testing of Android applications via model abstraction and refinement[C]//2019 IEEE/ACM 41st International Conference on Software Engineering, 2019: 269-280
- [10] SALIHU I A, IBRAHIM R, AHMED B S, et al. AMOGA: a static-dynamic model generation strategy for mobile apps testing [J]. IEEE Access, 2019, 7: 17158-17173
- [11] BEHRANG F, ORSO A. Test migration between mobile apps with similar functionality[C]//2019 34th IEEE/ACM International Conference on Automated Software Engineering, 2019: 54-65
- [12] PAN M, XU T, PEI Y, et al. GUI-guided test script repair for mobile apps[J]. IEEE Trans on Software Engineering, 2022, 48(3): 910-929
- [13] CRACIUNESCU M, MOCANU S, DOBRE C, et al. Robot based automated testing procedure dedicated to mobile devices[C]// 2018 25th International Conference on Systems, Signals and Image Processing, 2018: 1-4
- [14] MAO K, HARMAN M, JIA Y. Robotic testing of mobile APPS for truly black-box automation[J]. IEEE Software, 2017, 34(2): 11-16
- [15] BANERJEE D, YU K. Robotic arm-based face recognition software test automation[J]. IEEE Access, 2018, 6: 37858-37868
- [16] NASS M, ALÉGROTH E, FELDT R. Why many challenges with GUI test automation(will) remain[J]. Information and Software Technology, 2021, 138: 106625

- [17] DEKA B, HUANG Z, FRANZEN C, et al. RICO: a mobile app dataset for building data-driven design applications[C] // Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology, 2017: 845-854
- [18] BOCHKOVSKIY A, WANG C Y, LIAO H Y M. YOLOv4: Optimal speed and accuracy of object detection[J/OL].(2020-04-05) [2021-08-12]. <https://arxiv.org/abs/2004.10934>
- [19] LIU W, ANGUELOV D, ERHAN D, et al. SSD: single shot multibox detector[C] // European Conference on Computer Vision, Cham, 2016: 21-37
- [20] LIN T Y, GOYAL P, GIRSHICK R, et al. Focal loss for dense object detection[C] // Proceedings of the IEEE International Conference on Computer Vision, 2017: 2980-2988
- [21] CHEN C, SU T, MENG G, et al. From UI design image to GUI skeleton: a neural machine translator to bootstrap mobile GUI implementation[C] // Proceedings of the 40th International Conference on Software Engineering. 2018: 665-676
- [22] MORAN K, BERNAL-CÁRDENAS C, CURCIO M, et al. Machine learning-based prototyping of graphical user interfaces for mobile apps[J]. IEEE Trans on Software Engineering, 2018, 46(2): 196-221
- [23] MASCI J, MEIER U, CIRESAN D, et al. Stacked convolutional auto-encoders for hierarchical feature extraction[C] // International Conference on Artificial Neural Networks, Berlin, Heidelberg, 2011: 52-59

Visual judgment approach of isomorphic GUI for automated mobile app testing

XUE Feng¹, WU Junsheng², ZHANG Tao², WANG Wei², CHENG Jing³

(1.School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China;
2.School of Software, Northwestern Polytechnical University, Xi'an 710072, China;
3.School of Computer Science and Engineering, Xi'an Technological University, Xi'an 710021, China)

Abstract: Currently, the rapid growth of mobile apps requires automated testing technology to ensure their quality. Automated testing of mobile apps is usually closely related to the recognition and judgment of their graphical user interface (GUI), but there usually are numerous isomorphic GUIs with different styles and contents, and similar structure and function in mobile apps. In automatic testing, isomorphic GUI is easy to cause the issue of state space explosion, which leads to low efficiency or failure of testing. In view of the limitations of traditional automatic recognition of isomorphic GUI, this paper presents a GUI similarity judgment approach based on visual feature information. Firstly, the GUI component elements are identified by object detection, and then the GUI skeleton is abstracted. Secondly, the visual features of the GUI skeleton are extracted by a convolutional autoencoder. Finally, the isomorphic GUI judgment is completed by comparing the similarity of GUI visual features. The experimental results show that the proposed approach can effectively shield the influence of GUI style and content, complete the isomorphic GUI recognition more accurately and optimize the efficiency of automated mobile app testing.

Keywords: mobile app testing; GUI visual features; isomorphic GUI; GUI similarity judgment

引用格式:薛峰, 武君胜, 张涛, 等. 面向移动应用自动化测试的同构用户界面视觉判断方法[J]. 西北工业大学学报, 2022, 40(4): 804-811

XUE Feng, WU Junsheng, ZHANG Tao, et al. Visual judgment approach of isomorphic GUI for automated mobile app testing[J]. Journal of Northwestern Polytechnical University, 2022, 40(4): 804-811 (in Chinese)