arXiv:2103.07085v1 [cs.SE] 12 Mar 2021

**19**

**通过图像自动编码器进行基于线框的 UI 设计搜索**

JIESHAN CHEN，澳大利亚国立大学，澳大利亚

杨春陈\* ，澳大利亚莫纳什大学

ZHENCHANG XING \* ,澳大利亚国立大学，澳大利亚

XIN XIA ，澳大利亚莫纳什大学

LIMING ZHU \* , Data61,CSIRO, 澳大利亚

JOHN GRUNDY ，澳大利亚莫纳什大学

王金水\* , 福建工业大学, 中国

UI设计是软件开发不可或缺的一部分。对于很多没有太多 UI 设计经验的开发人员来说，将他们暴露在一个庞大的真实应用 UI 设计数据库中，可以帮助他们快速建立对软件功能设计空间的真实理解，并从现有应用程序中获得设计灵感。然而，现有的基于关键字、基于图像相似性和基于组件匹配的方法无法在大型数据库中可靠地找到与开发人员绘制的 UI 线框图相关的高保真 UI 设计，因为面对 UI 设计的巨大变化. 在本文中，我们提出了一个基于深度学习的 UI 设计搜索引擎来填补这一空白。我们搜索引擎的关键创新是使用大型真实应用程序 UI 设计数据库训练线框图像自动编码器，无需标记相关的 UI 设计。我们实现了我们的 Android UI 设计搜索方法，并对人工创建的相关 UI 设计和 UI 设计搜索结果的人工评估进行了广泛的实验。我们的实验证实了我们的搜索引擎优于现有的基于图像相似性或组件匹配的方法，并证明了我们的搜索引擎在现实世界 UI 设计任务中的有用性。

CCS 概念： •**软件及其工程**^**软件库和存储库**；**软件原打字**;

其他关键词和短语：Android、UI 搜索、深度学习、自动编码器。

**ACM 参考格式：**

陈洁善、陈春阳、邢振昌、夏欣、朱立明、约翰·格兰迪、王金水。2020. 通过图像自动编码器进行基于线框的 UI 设计搜索。*ACM 翻译。软件 英。方法论。*29、3，第十九条（2020 年 5 月），31页。<https://doi.org/10.1145/3391613>

\*通讯作者。

t也有 Data61、CSIRO。

$也与新南威尔士大学合作。

作者地址：[Jieshan](mailto:Jieshan.Chen@anu.edu.au) Chen, [Jieshan.Chen@anu.edu.au](mailto:Jieshan.Chen@anu.edu.au)，澳大利亚国立大学，澳大利亚；陈春阳，[Chunyang.Chen@](mailto:Chunyang.Chen@monash.edu) monash.edu，澳大利亚莫纳什大学；邢振昌，澳大利亚国立大学，[Zhenchang.Xing@anu.edu.au](mailto:Zhenchang.Xing@anu.edu.au)；Xin Xia，[Xin.Xia@monash.edu](mailto:Xin.Xia@monash.edu)，澳大利亚莫纳什大学；Liming Zhu, Liming.Zhu@data61.csiro.au, Data61,CSIRO, 澳大利亚；John Grundy，[John.Grundy@monash.edu](mailto:John.Grundy@monash.edu)，澳大利亚莫纳什大学；[Jinshui](mailto:ymkscom@gmail.com) Wang，[ymkscom@gmail.com](mailto:ymkscom@gmail.com)，福建工业大学，中国。

允许将本作品的全部或部分制作数字或硬拷贝以供个人或课堂使用，但不收取任何费用，前提是拷贝不是为了盈利或商业利益而制作或分发的，并且拷贝带有此通知和第一页的完整引文. 必须尊重 ACM 以外的其他人拥有的本作品组件的版权。允许使用信用进行抽象。以其他方式复制或重新发布、在服务器上发布或重新分发到列表，需要事先获得特定许可和/或费用。从[permissions@acm.org](mailto:permissions@acm.org)请求[权限](mailto:permissions@acm.org)。

© 2020 计算机协会。

1049-331X/2020/5-ART19 $15.00

<https://doi.org/10.1145/3391613>

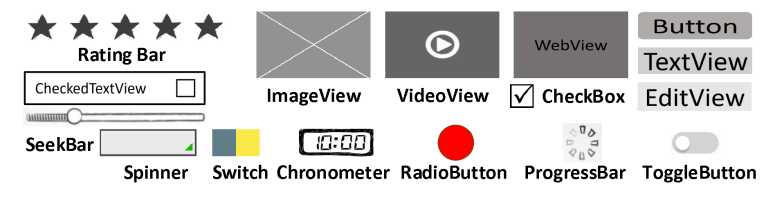
1 介绍

图形用户界面 (GUI) 在现代桌面软件、移动应用程序和 Web 应用程序中无处不在。它提供了软件应用程序与其最终用户之间的可视化界面，通过该界面他们可以相互交互。设计良好的GUI，使一个应用程序简单，实用和有效地使用，这显著影响应用的成功和它的用户[忠诚度7 ， 46 ，71 ]。例如，在竞争激烈的移动应用程序市场中，应用程序的 GUI 甚至图标的设计对于将应用程序与竞争对手区分开来、吸引用户下载、减少用户投诉和留住用户变得至关重要[[ 4 ,](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark87) 37 , 56 ]]。

设计 GUI 的视觉组合是软件开发的一个组成部分。根据最初的用户需求和软件需求，设计人员通常首先通过选择具有特殊功能的高度简化的可视化组件（例如图1所示的那些）来设计所需GUI的*线框*) 并确定所选组件的布局，以支持适合应用程序数据的交互和实现用户目标所需的操作，并通过与现有在线设计示例进行比较来迭代修改其设计。然后，他们为 GUI 组件添加高保真视觉效果，例如颜色和排版，并将特定于应用程序的文本和图像添加到 GUI 设计中。当然，线框设计和高保真GUI设计是相互交织和迭代的，设计师通过去除不必要的视觉组件、添加缺失的组件、优化组件的布局和视觉效果来不断探索设计空间。

为了满足用户的需求，设计一个好的 GUI 不仅需要设计原则和指南的具体知识（例如，Android Material Design [ 5 ]、iOS Human Interface Guidelines [ 10 ]），还需要对设计空间的理解，这些知识具有可以潜在使用的视觉组件、它们的布局选项和视觉效果选择的巨大变化。如图2所示，GUI 的设计空间，即使是简单的注册功能，也可能非常大。然而，由于 UI 设计师的短缺 [ 42]，对UI设计空间不太了解的软件开发者，往往不得不在软件开发中扮演设计师的角色，尤其是在初创公司和开源项目中。例如，在对超过 5,700 名开发人员的调查中[[ 1 ]](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark82)，51% 的受访者表示他们没有接受太多 UI 设计培训，但他们从事 UI 设计任务，比其他开发任务更多。实际上，在开发应用程序时，软件开发人员和设计人员所关注的内容是完全不同的。开发人员试图使应用程序正常工作，而设计人员的目标是让它变得可爱 [ 12]，这使得软件开发人员很难直接作为设计师工作。需要一种有效的机制来支持此类开发人员在其 UI 设计工作中探索和了解 UI 设计空间。

为开发者提供一个UI设计搜索引擎来搜索现有的UI设计，可以帮助开发者快速建立对GUI设计空间的真实理解，并从现有应用中获取灵感，用于自己应用的UI设计。然而，与国内支持的代码搜索[对比55 ，62 ，63 ，70 ]，出现了针对UI设计的搜索支持很少。现有的 UI 设计搜索方法 [ 6 , 11 , 22] 基于描述软件功能、UI 设计模式或 GUI 组件的关键字。虽然基于关键词的 UI 搜索可以提供一些初步的设计灵感，但仍然需要一个更先进的 UI 设计搜索引擎更有针对性地探索设计空间，它可以直接将示意性 UI（例如，线框）作为输入开发人员绘制草图并返回与输入类似的高保真 UI 设计（有关激励方案，请参见第[2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM" \l "bookmark13" \o "当前文件)节）。然而，一些关键字很难描述所需 UI 设计的视觉语义，例如使用的视觉组件及其布局。



1. Android UI 设计中最常见的线框组件。每个组件都有自己的功能。例如，TextView 将向用户显示文本，而 EditText 允许用户输入文本。
2. 由于高级 UI 设计搜索可以将 UI 设计视为图像，因此一种简单的解决方案可能是通过某些图像特征的图像相似性来搜索 UI 设计，例如颜色直方图 [ 45 ] 或尺度不变特征变换 (SIFT) [ 54 ]。虽然这样的图像特征在测量图像相似性时，它们对 GUI 的视觉语义（视觉组件及其组成）具有竞争性。因此，图像方面类似的 UI 设计很可能在设计方面无关紧要。或者，可以试探性地匹配基于它们的类型，位置和尺寸的单个可视化组件用于测量的两个UI设计[相似度19 ，66 ，78 ]。然而，这种方法受到预定义的组件匹配规则的限制，并且对截止匹配阈值很敏感。此外，单个组件匹配试探法通常会检索许多不相关的 UI 设计，因为单个组件匹配无法有效地将 GUI 中组件的视觉组合作为一个整体进行编码。

最相关的工作是 Rico，它引入了一个新的 UI 数据集，讨论了五种潜在的用法并演示了辅助 UI 搜索的可能性。他们的数据集是通过自动应用程序探索和通过招募人群工作人员手动探索来收集的。在 UI 搜索演示方面，他们使用了一个只有 6 个全连接层的简单多层感知器，将 UI 组件简化为文本/非文本，并展示了几个示例，没有任何性能结果的详细统计。如上所述，UI 设计是复杂的，有多种变体，因此不可能仅使用文本和非文本来表达一种 UI 设计的核心概念。此外，具有任意数量和位置的不同小部件的复杂组合在类型学方面显示了巨大的设计空间。所以，具有高度简化的小部件的幼稚方法不足以解决此任务。请注意，Rico 评估仅显示了几个示例，没有对检索准确性、数据问题、模型限制、失败案例和有用性评估进行任何详细研究。因此，我们无法知道他们模型的泛化或性能。

在本文中，我们提出了一种使用卷积神经网络而不是纯全连接层来开发基于深度学习的 UI 设计搜索引擎的方法。为了向开发人员展示各种软件功能的多样化、真实应用程序 UI 设计，我们使用逆向工程方法（例如 [ 27] 中的自动 GUI 探索方法）]) 来构建现有应用程序的 UI 屏幕截图（及其相应的线框）的大型数据库。我们通过分析几个设计平台和 UI 实现细节，进一步确定了 16 个用户交互组件，它们缩小了设计人员和开发人员之间的差距。我们的方法执行基于线框的 UI 设计搜索。线框捕捉视觉组件的类型和布局信息，但忽略了它们的高保真视觉细节。因此，它们可以以最少的努力快速原型化和改进，并且可以检索视觉上不同但语义相关的 UI 设计（参见图2例如）。我们的方法不执行单个组件匹配，而是尝试判断整个 UI 设计的相关性。开发如此强大的 UI 设计相关模型的一个关键挑战是不存在标记的相关 UI 设计，并且需要大量的手动工作来注释如此大的数据集。因此，我们不能使用像 [ 29 , 50 ]这样的监督学习方法来训练模型来编码 UI 设计的视觉语义。为了克服这一挑战并减轻繁重的手动工作，我们设计了一个线框自动编码器，可以以无监督的方式使用大型 UI 线框数据库进行训练。经过训练后，该自动编码器可以对ACM Trans进行编码。软件 英。方法论，卷。29、3、19条。发表日期：2020年5月。

用户的查询线框和现有应用程序的 UI 截图，通过它们在 UI 设计向量空间中的相应线框。在这个向量空间中，通过 k-最近邻 (kNN) 搜索可以轻松地检索与查询线框类似的 UI 屏幕截图。

作为概念证明，我们实现了在 Google Play中 7,748个下载次数最多的 Android 应用程序的25 个类别[[[1]](#footnote-1)[1]](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark0)的54,987 个 UI 屏幕截图的数据库中搜索 Android 移动应用程序 UI 设计的方法。我们评估我们的UI设计的搜索ENGIN的性能，通用性和实用性[ë [[2]](#footnote-2)[2]](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark1) [[[3]](#footnote-3)](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark0) [[[4]](#footnote-4)](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark1) 通过对组件缩放和组件移除操作生成的 4500 对相关 UI 设计的自动评估，对来自 25 个应用程序（不在我们的数据库中）的 50 个未见过的查询 UI 返回的前 10 个 UI 设计的相关性进行人工评估，以及对 18 位非专业 UI 设计师进行的五项 UI 设计任务的用户研究。我们的评估证实了我们的方法比基于低级图像特征（颜色直方图和 SIFT）、单个组件匹配启发式和基于全连接层的神经网络的基线具有更高的性能。用户研究参与者高度赞赏我们的工具在协助他们的设计工作时 UI 设计搜索结果的相关性、多样性和实用性。他们还指出了用户对 UI 设计搜索的几个需求，例如约束感知的 UI 设计搜索，

我们的贡献可以总结如下：

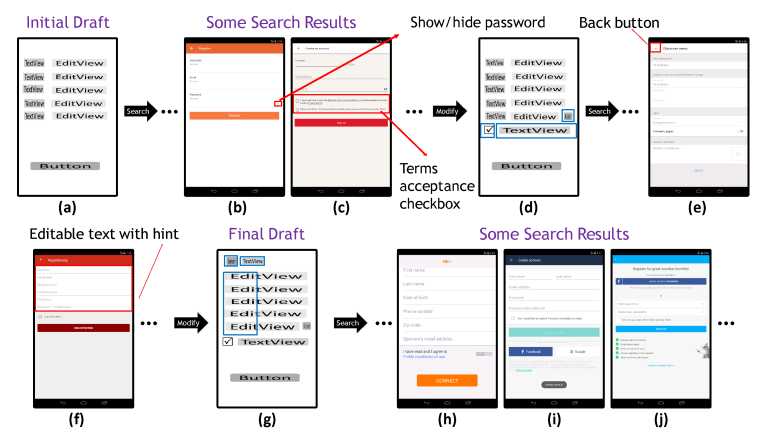
* 我们提出了一种新的基于深度学习的方法，以无监督的方式使用卷积神经网络来构建 UI 设计搜索引擎，该引擎在 UI 设计的巨大变化面前灵活而稳健。
* 我们通过探索不同的线框图方法，构建了下载次数最多的Android 应用程序UI 设计的大型线框图数据库，并开发了一个基于Web 的搜索界面来实现我们的方法。
* 我们广泛的实验证明了我们的方法和工具支持的性能、通用性和实用性，并指出了有趣的未来工作。

本文的其余部分组织如下。第[2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark13)节提出了一个激励方案来描述我们工具的潜在用途。第[3](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark16)节描述了数据收集的一般方法、模型结构和我们的 UI 设计搜索引擎的原理。第[4](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark31)节揭示了选择线框类型和表示、模型超参数选择和工具实现的详细方法。我们分别在第[5](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark40)节和第[6](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark6)节中描述了我们的实验设置和结果，就线框的最佳表示、准确性、泛化性和实用性而言。第[7](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark67)节讨论了对有效性的潜在威胁[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark67)部分[第 8](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark74)节讨论相关工作，第[9](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark77)节总结文章。

2 激励情景

一家初创公司需要为其移动应用程序设计 UI。像许多小公司一样 [ 42 ]，由于预算限制，它没有专业的 UI 设计师。所以设计工作被分配给了软件开发人员露西。Lucy 有一些桌面软件前端开发经验，但从未设计过移动应用程序 UI。

Lucy 的第一个任务是设计一个注册 UI，用于在用户注册期间收集用户信息，例如用户名、密码和电子邮件。根据她之前的桌面软件开发经验，Lucy 设计了一个非常基本的注册 UI（图2 (a)）。它有几个并排的 TextView



1. UI 设计搜索：好处和挑战

EditView：TextView 用于显示要收集的信息的标签，EditView 用于输入信息。在底部，它有一个按钮，用于提交输入的用户信息。

Lucy 担心她的设计不完整，也不适合移动应用程序。她想看看其他应用程序是否设计了和她一样的注册 UI，但她不想为了看到他们的注册 UI（如果有）而从应用市场随机下载和安装应用程序。它不仅耗时，而且无法系统地查看相关的 UI 设计。更好的解决方案是将她的 UI 设计输入一个有效的 UI 设计搜索引擎，该引擎可以从大型 UI 设计数据库中返回相似但视觉效果不同的 UI 设计。

Lucy 尝试使用这样的 UI 设计搜索引擎来获取与她最初注册的 UI 设计相似的 UI 设计列表。观察返回的 UI 设计，Lucy 意识到虽然她的初始设计具有基本功能，但确实遗漏了一些不错且重要的功能。例如，她可以添加一个显示/隐藏密码按钮（如图2 （b）），方便用户确认输入的密码。此外，注册 UI 是用户访问和确认相关条款和条件的好地方（例如，图2 (c)）。基于这些观察，Lucy 将她的设计改进为图2 (d) 中的设计（蓝色框中突出显示的更改）并再次搜索 UI 设计数据库。

通过观察搜索结果，Lucy 对流行的注册表单的需求有了真实的了解，包括视觉组件、布局选项和视觉效果，并进一步完善了她的设计。例如，移动应用程序通常在顶部有一个导航按钮（例如，图2 （b）/（c）/（e）/（f）中的后退按钮）以方便 UI 页面之间的导航。此外，与桌面软件中传统的并排标签文本输入设计不同，移动应用程序使用带有提示的可编辑文本来实现相同的效果（图2 （c）/（e）/（f））。这种设计更适用于屏幕比台式电脑小得多的移动设备。基于这些设计灵感，Lucy 进一步完善了她的设计，如图2所示（G）。将 UI 设计搜索结果（（例如，图2 (h)/(i)/(j)）与图2 (g) 中的设计进行比较，Lucy 现在对她的最终 UI 线框图充满信心。此外，观察到许多相关和多样化的 UI 设计为 Lucy 为她的 UI 设计高保真视觉效果（例如，颜色系统、排版）提供了许多灵感。

ACM 翻译。软件 英。方法论，卷。29、3、19条。发表日期：2020年5月。

1. 我们方法的概述

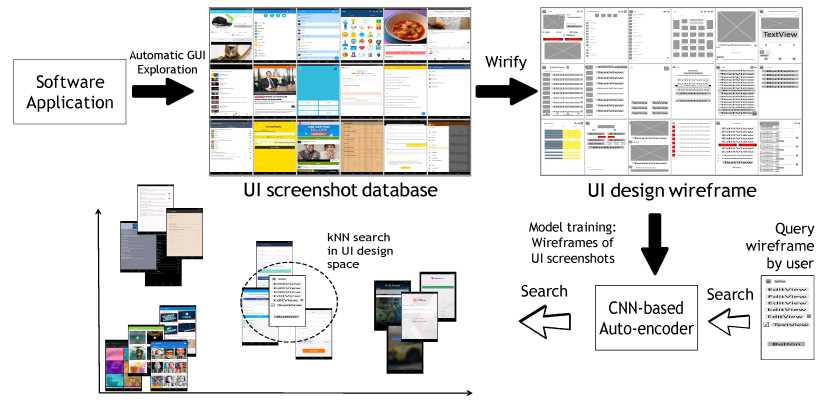
如图2 中的UI 设计所示，GUI 的设计空间可能非常巨大，在以下方面存在很大差异：(1) 所使用的可视化组件的类型（例如，许多 UI 设计中的复选框与图2 中唯一的开关（h )); (2) 设计中视觉组件的数量（例如，图2 (g)、(i) 和 (j) 中的可编辑文本和按钮）；(3) 视觉组件的位置和大小（例如，图2 (b) 与 (h) 中的可编辑文本和按钮）；(4) 可视化组件的布局（例如，图2 (a) 中的并排标签文本输入与图2 (b) 中的上下标签文本输入，或图2 中的leftcheckbox + right-text）(c) 与图2 (h) 中的左文本 + 右切换。实现 UI 设计搜索的上述设想的好处，需要搜索引擎在面对巨大的变化时具有灵活性和健壮性，并在 UI 设计的相似性和变化之间取得良好的平衡。

#### 3.我们的方法

图3概述了我们基于深度学习的方法，用于构建 UI 设计搜索引擎，该引擎在 UI 设计的巨大变化面前灵活而稳健。我们的方法包括三个主要步骤：1）使用基于GUI自动勘探方法建立多元化，实时应用程序的UI设计（第一个大型数据库[3.1 ）](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark19) ; 2) 训练基于 CNN 的线框自动编码器，使用大型 UI 设计线框数据库对 UI 设计的视觉语义进行编码（第[3.2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark25)节[）](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark25)；和 3) 使用经过训练的线框编码器将 UI 设计嵌入到潜在向量空间中，并支持基于线框的 kNN UI 设计搜索（第[3.3](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark28)节[）](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark28)

##### 3.1真实应用 UI 设计的大型数据库

需要一个包含各种不同软件功能的真实应用程序 UI 设计的大型数据库，以便向开发人员展示真实的 UI 设计空间。为此，我们采用自动数据收集方法，首先从现有应用程序构建一个大型 UI 设计数据库，然后使用收集到的数据进一步构建我们的线框数据集。

* + 1. *自动 GUI 探索。*可以使用不同的技术来自动探索移动应用程序、web应用程序或桌面应用程序的gui。尽管技术细节不同，但这些技术在概念上以相同的方式工作。**它们通过模拟用户与应用程序的交互来自动探索应用程序的GUI，并输出GUI截屏图像和运行时可视化组件信息，这些信息识别每个组件的类型和在截屏中的坐标。**在 GUI 探索过程中，可能会重复访问相同的 GUI，但会丢弃重复的截图，以确保收集的 UI 设计的多样性。为了提高收集到的 UI 设计的质量，可以进一步启发式**过滤掉无意义的 UI**，例如，移动设备的主屏幕，没有太多设计的简单 UI，如只有一个图像组件的 UI。具体来说，我们首先从 Google Play 抓取应用程序，并在模拟器中自动安装和运行应用程序。对于每个应用程序，我们的模拟器通过模拟用户的操作（包括单击按钮、输入文本和滚动屏幕）与应用程序进行交互。当进入一个新页面时，我们的工具会截取当前 UI 的屏幕截图并转储 XML 运行时代码。XML 运行时代码包含有关当前 UI 的所有信息，包括所有包含的组件及其相应的边界、类、文本、关于可执行性的布尔属性（例如可检查、可点击和可滚动）以及它们之间的层次关系。为了确保我们探索的 UI 的覆盖范围，我们还应用了 [ XML 运行时代码包含有关当前 UI 的所有信息，包括所有包含的组件及其相应的边界、类、文本、关于可执行性的布尔属性（例如可检查、可点击和可滚动）以及它们之间的层次关系。为了确保我们探索的 UI 的覆盖范围，我们还应用了 [ XML 运行时代码包含有关当前 UI 的所有信息，包括所有包含的组件及其相应的边界、类、文本、关于可执行性的布尔属性（例如可检查、可点击和可滚动）以及它们之间的层次关系。为了确保我们探索的 UI 的覆盖范围，我们还应用了 [27 ]，它定义了每个潜在可操作组件被按下的概率（或权重）。规则定义为：（1）频率较高的动作被赋予较低的权重，因为我们需要给其他罕见的动作机会执行；(2) 导致更多后续 UI 的动作将具有更高的权重，以便探索各种 UI；(3) 控制一些特殊的动作（如*硬件后退*和*滚动）*，以防它们在错误的时间关闭当前页面或影响他人的动作。每个可执行组件的实际权重由*weights(a) = (a* \* *T a* + *0* \* *C a )/y 给出*\* *F a*，其中*a、T a*、*C a* 分别为动作、不同类型动作的权重和当前UI 中未探索的可执行组件的数量，*a、0、y*为超参数。由于此收集过程是自动的，因此可能会多次重新访问某些 UI，我们需要删除重复数据。为此，我们通过比较 GUI 组件序列的哈希值，将当前转储的 XML 代码文件与收集的数据进行比较。

*3.1.2 接线。*我们的方法执行基于线框的 UI 设计搜索。因此，不同于现有的逆向工程方法，我们需要进一步为数据库中收集的每个UI截图获取一个UI线框图。为此，我们有两个步骤。首先，我们通过分析流行的设计师工具和 UI 的底层实现细节，在设计级别定义了一组对于不同类型的用户交互必不可少的线框组件。其次，我们通过探索实验找到了每个组件的“正确”表示。

**组件类型的选择。**首先，Android UI组件在功能上有两种类型：布局组件和UI控件组件[ 13]]。UI控件组件（如button、textView、ImageView）是我们可以看到并与之交互的可见组件，而布局组件（如linearLayout、relativeLayout）用于约束UI控件组件之间的位置关系。由于本作品的输入只是线框图，涉及到的控件组件选择较多，位置粗略，所以我们只关注UI控件组件（我们简称为UI组件）。我们的数据集中有 21 个 UI 组件，我们选择它们作为线框的候选组件。在转换带有相应运行时代码的 UI 截图时，我们考虑两个因素： (1) 设计线框时功能相似且视觉效果相似的组件不会有太大差异；(2) 很少使用的组件对于 UI 设计可能不是很有用。首先，我们将 MultiAutoCompleteTextView 与 EditText 合并。它们都启用可编辑文本，但 MultiAutoCompleteTextView 具有额外的文本自动完成功能。不过这个功能可以在 EditText 中通过显化底层后台代码来实现。我们还将 ImageButton 与 Button 合并，因为它们都使用户可以单击它们然后触发一些

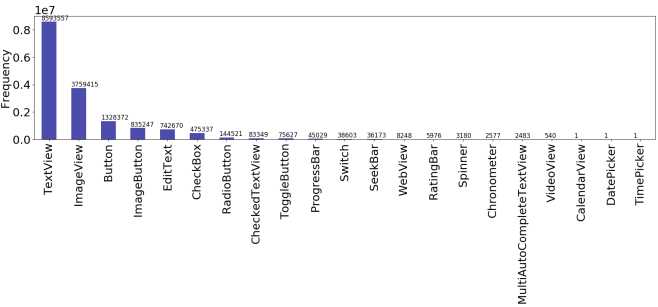


图 4. 包含在我们数据集中的每个组件的频率的条形图

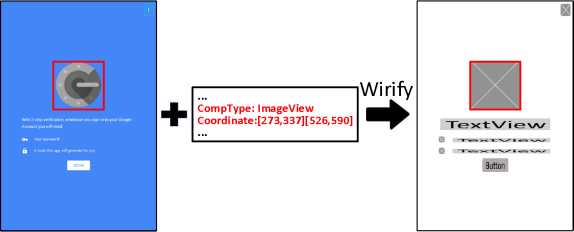


图 5. 线框的视觉渲染说明了顶部的大 ImageView。

事件。对于第二个考虑，我们忽略了 CalenderView、TimePicker 和 DatePicker 组件，因为它们在我们的数据集中只出现一次（见图4 ）。低频可能是因为它们进一步分离成几个子组件，例如TextView和Spinner。因此，我们留下 16 个组件作为我们最终的线框单元集。这 16 种类型的组件也被用于移动 UI 设计的流行线框工具广泛覆盖，如 Adob​​e XD [ 14 ]、Fluid UI [ 16 ]、Balsamiq Mockups [ 15]]。线框图的实现过程中，我们在线框中用EditText的表示来表示MultiAutoCompleteTextView，用Button的表示来表示ImageButton。由于上述原因，我们没有在线框中绘制 CalenderView、TimePicker 和 DatePicker。对于其他组件，我们在线框中使用它们自己的表示来绘制它们。我们发布的线框转换的源代码，在我们公司网址[ë [[5]](#footnote-5)[3] 。[[6]](#footnote-6)](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark2)

**接线过程。**在定义了 16 个核心线框组件后，然后使用我们在自动探索应用程序期间转储的 XML 运行时代码文件将 UI 屏幕截图连接到 UI 线框。请注意，在此过程中没有不确定性，因为它完全是基于规则的过程。我们根据直接从 Android 操作系统转储的运行时代码对屏幕截图进行线框化，其中包含 UI 屏幕截图中每个组件的类型和坐标。因此，这些UI截图和对应的运行时代码文件是完美匹配的，线框转换时不会出现错误。图[5](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark24)说明了这一点

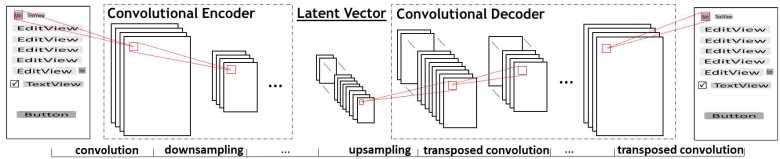


图 6. 我们的线框自动编码器的架构

高级线框化过程：UI线框与UI截图大小相同，并有一个白色画布，在该画布上绘制线框组件与UI截图中每个对应的视觉组件相同的位置和相同的大小（例如，ImageView）。但是，线框组件会忽略相应视觉组件的颜色和文本/图像内容。

**线框最佳表示方式的探索。**除此之外，我们需要定义这些组件的表示来构建我们最终的线框数据集。我们不使用流行工具的默认图像 [ 14 - 16] 因为它们不够精确或一般。相反，我们用不同颜色的简单矩形表示它们，这可以明确地告诉模型这些组件是不同的。由于设计空间巨大，考虑所有颜色和调色板是不现实的，因此我们考虑三种典型的变体来表示这些视觉组件，即不同的灰度值、不同的颜色以及不同纹理的不同颜色。组件的最佳表示的详细探索设置和结果将分别在第[5.2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark45)节和第[6.1](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark7)节中讨论。请注意，为了避免潜在的分心，我们将线框呈现为具有不同灰度颜色背景的文本，以帮助读者更好地理解论文中的这些线框。

* 1. 基于CNN的线框自编码器

确定 UI 设计的相关性是一项具有挑战性的任务，因为它不仅需要对单独的视觉组件进行编码，还需要对整个 UI 中组件的视觉组合进行编码。使用哪些组件以及如何在 UI 中组合它们的设计空间是巨大的，因此无法启发式枚举。基于 CNN 的模型可以从大型图像数据库中自动学习潜在特征，在许多计算机视觉任务中都优于手工制作的特征。虽然我们有一个庞大的 UI 设计数据库，但这些 UI 设计的相关性是未知的。因此，我们必须训练一个 CNN 模型，以无监督的方式对 UI 设计的视觉语义进行编码。为此，我们选择使用基于 CNN 的图像自动编码器架构，该架构只需要一组未标记的输入图像进行模型训练。如图6所示，我们的自动编码器将 UI 线框图像作为输入。它有两个组成部分：编码器通过卷积和下采样层将输入线框压缩成潜在向量表示，然后解码器通过上采样和转置卷积层从这个潜在向量表示重建输出图像。重建的输出图像应尽可能与输入图像相似，这表明潜在向量从输入线框设计中捕获信息特征以进行重建。然后可以使用 UI 设计的这种潜在向量表示来衡量 UI 设计的相关性。

*3.2.1 卷积。*卷积运算对图像执行线性变换，使得不同的图像特征变得显着。根据 CNN 可视化的研究 [ 52 , 76 ]，浅卷积层检测简单的特征，例如边缘、颜色和形状，然后在深卷积层中组合以检测特定领域的特征（例如，UI 的视觉语义）我们工作中的设计）。

图像表示为像素值矩阵，即 0 ^ *p h0 d* ^ 255，其中*h、w*和*d*是图像的高度、宽度和深度。d = 1 表示灰度图像，d = 3 表示 RGB 彩色图像。图像的卷积使用一个*核* （即像 3 X 3 X d这样的小矩阵的可学习参数），并将内核滑过图像的高度和宽度一次 1 个像素。在每个位置，卷积操作将内核元素与图像的内核大小子区域相乘，并将这些值相加为输出值。转置卷积与普通卷积相反。它将一个值与一个内核相乘并输出一个内核大小的矩阵。一个卷积层可以应用多个内核 ( n )。卷积层之后的输出矩阵（h X w X n ）称为特征图，可以将其馈入后续网络层进行进一步处理。每个内核映射h X w特征图中对应一个核，可以看成是突出显示了一些特定特征的图像。

*3.2.2 下采样和上采样。*在编码器中，下采样（也称为池化）层将前面卷积层的输出特征图作为输入，并产生空间（高度和宽度）缩减的特征图。下采样层由池单元网格组成，每个单元汇总输入内核映射的大小为*z Xz*的区域。由于下采样层在每个输入内核图上独立运行，因此输出特征图的深度与输入特征图的深度保持一致。在我们的架构中，我们采用*1-maxpooling* [ 60 ]，它取z X z 中的最大值（即最显着的特征）地区。1-max pooling 带来了图像移动、旋转和缩放不变性的好处，导致在 UI 设计中对编码组件空间变化有一定程度的不敏感。

在解码器中，我们使用与下采样相反的上采样层。它们通过用多个值替换输入特征图中的每个值来增加特征图的空间大小（高度和宽度）。在我们的架构中，我们采用最近邻插值[ 49 ] ，即将特征图中的原始像素丰富为与原始像素具有相同值的z X z区域。上采样层逐渐增加特征图的空间大小，直到解码器最终重建与输入线框大小相同的输出线框。

*3.2.3 模型训练。*编码器和解码器作为端到端系统进行训练。给定 UI 线框图像*X*，编码器将其压缩为潜在向量*V* : 0 : *X ^ V*其中0表示编码器的卷积层和下采样层的功能。然后解码器将潜在向量V解码为输出线框图像T : ^ : V ^ T其中^表示解码器的上采样和转置卷积层的功能。目标是最小化输入线框X之间的差异和输出线框T : argmin^ ||X - T || 2 . 我们训练我们的线框自动编码器以最小化均方误差 (MSE) [ 21 ]的重建误差，即*L(X,* T ) = ||X - T || 2 . 在训练时，我们使用随机梯度下降 [ 23 ]优化训练数据集上的 MSE 损失。解码器将误差差异反向传播到其输入，即编码器，允许我们使用未标记的输入线框来训练线框编码器。

* 1. UI 设计空间中的 kNN 搜索

如图2所示，通过训练线框自动编码器，我们获得了一个卷积编码器，可以将输入线框编码为潜在向量表示。给定一个自动收集的 UI 截图数据库（可以与用于模型训练的 UI 截图不同），我们使用这个经过训练的线框编码器将 UI 截图通过相应的线框嵌入到 UI 设计空间*S 中*。每个 UI 屏幕截图*uis*都表示为一个潜在的

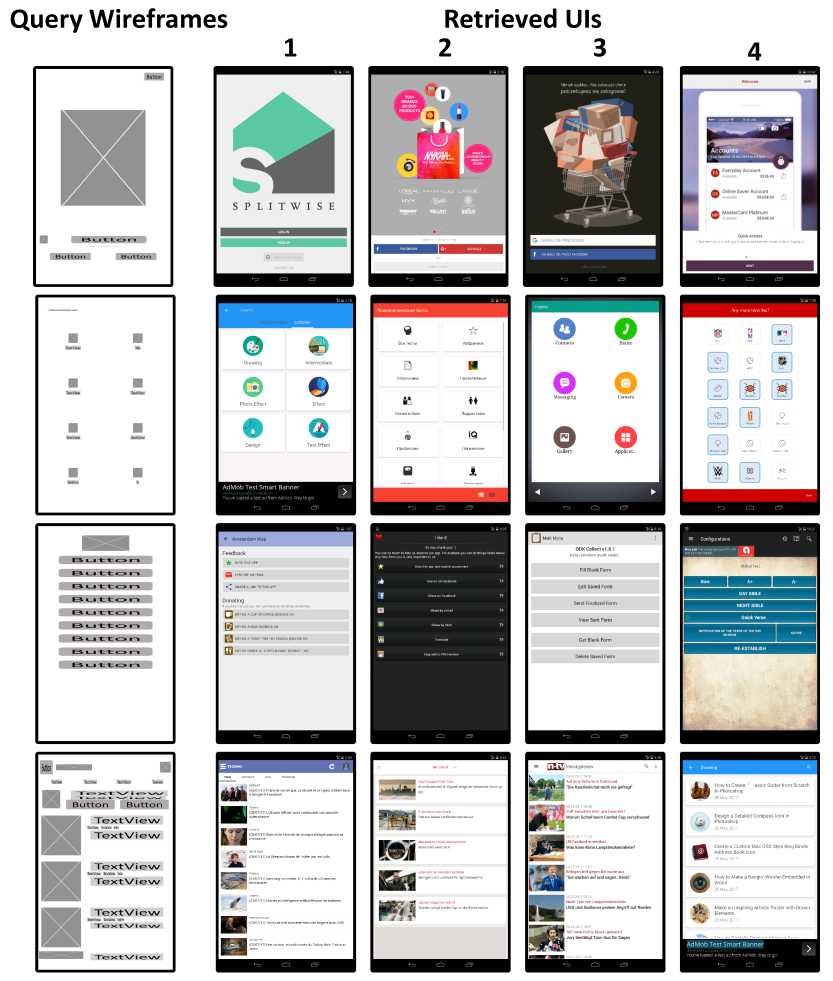


图 7. UI 设计空间中 kNN 搜索的示例

此 UI 设计空间中的向量*V(uis* ) 。由于查询线框*的\**用户绘制，我们还利用训练的线框编码器嵌入*的\**成矢量V *（的\* ）*在UI的设计空间。然后，我们在 UI 设计空间中执行 k-最近邻 (kNN) 搜索，以找到嵌入与查询线框最相似（通过本工作中的均方误差 (MSE)）的前 k 项的 UI 屏幕截图*uis*，即精氨酸。/seS || V *(uis* ) , V *( \** )|| 2 . 图7 展示了一些 UI 设计ACM Trans 的例子。软件 英。方法论，卷。29、3、19条。发表日期：2020年5月。

我们的实证研究结果。第四个例子表明我们的模型可以成功地编码相当复杂的 UI 设计的视觉语义。

1. 概念验证实施

4.1 数据收集

我们开发了一个用于搜索 Android 移动应用程序 UI 设计的概念验证工具。后端 UI 设计空间包含来自 7,748 个 Android 应用程序的 54,987 个 UI 截图，属于 25 个应用程序类别。我们从 Google Play 抓取下载次数最多的 Android 应用程序，因为研究表明应用程序的下载数量与应用程序 GUI 设计的质量呈正相关 [ 37 , 56 ]。Android 应用程序分为三种类型：原生应用程序、混合应用程序和网络应用程序 [ 43]。这些类型的底层实现是不同的。本机应用程序使用 Android 本机小部件或从它们派生的小部件，混合应用程序使用 WebView 将其 HTML/CSS 部分组件编码为 Android 应用程序，而 Web 应用程序直接使用 HTML/CSS/JavaScript。在我们的论文中，我们只从原生和混合应用程序中收集 UI，因为它们易于从应用商店下载和安装，而 Web 应用程序没有这样的“应用商店”。我们删除了一些 UI，它们的 WebView 占据了屏幕的一半。我们保留小的 WebView 组件，因为它们大部分是广告

我们使用 [ 27 ] 中的自动 GUI 探索方法从这些 Android 应用程序中构建了一个大型 UI 屏幕截图数据库，详细过程在第[3.1.1](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark22)节中说明[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark22)我们总共从 Google Play 中抓取了 8,000 个安装数量最高的 Android 应用，并成功运行了 7,748 个 Android 应用并收集了 54,987 个 UI 屏幕截图。请注意，由于需要额外的硬件支持或我们的模拟器中缺少某些第三方库，一些应用程序被丢弃了。每个应用程序的 UI 屏幕截图的中位数为 3。我们的数据库包含非常多样化的 UI 设计（有关一些随机选择的示例，请参见图3 ）。更多示例可以在我们的 Github 存储库[y [[7]](#footnote-7)[4] 中](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark3)看到[[[8]](#footnote-8).](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark3)

4.2 模型超参数

我们工具中的线框自动编码器配置如下。输入线框是 RGB 彩色图像并缩放到 180 X 228 以进行高效处理。编码器使用了四个卷积层，分别使用了 16 个 3x3x3 内核、32 个 3x3x16 内核、32 个 3x3x32 内核和 64 个 3x3x32 内核。每个卷积层之后是一个*ReLU(x* ) = *max* ( 0 ,1 )非线性激活函数、一个具有 2 X 2 池化区域的 1-max 池化层和一个批量归一化层 [ 44 ]。解码器将输入内核映射中的值上采样为 2 X2 该值的区域。它有四个上采样层。在每个上采样层之后，解码器使用转置卷积层，分别使用32个3x3x64内核、32个3x3x32内核、16个3x3x32内核、3个3x3x16内核。

4.3 工具实现

我们在所有实验和工具[l [[9]](#footnote-9)[5] 中[[10]](#footnote-10)](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark4)使用 k=10 表示 KNN [。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark4) 图8显示了我们工具的前端。这个搜索界面的演示视频是我们的Github的复位器可用[Ÿ 4，](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark3)这表明我们的激励方案的UI设计搜索过程。使用我们的工具，用户在左侧画布上绘制 UI 线框。该工具目前支持 16 种最常用的线框组件类型，如图1所示。我们通过调查 Android GUI 框架和流行的 UI 设计工具（例如 Adob​​e XD [ 14 ]、Fluid UI [ 16 ]、Balsamiq Mockups [ 15 ]）将这些线框组件确定为 Android 移动应用程序的核心] 如第[3.1.2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark23)节所述[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark23)一旦用户点击搜索按钮，系统就会返回 UI 设计空间中的前 10 个（即 KNN 的 k=10）UI 设计

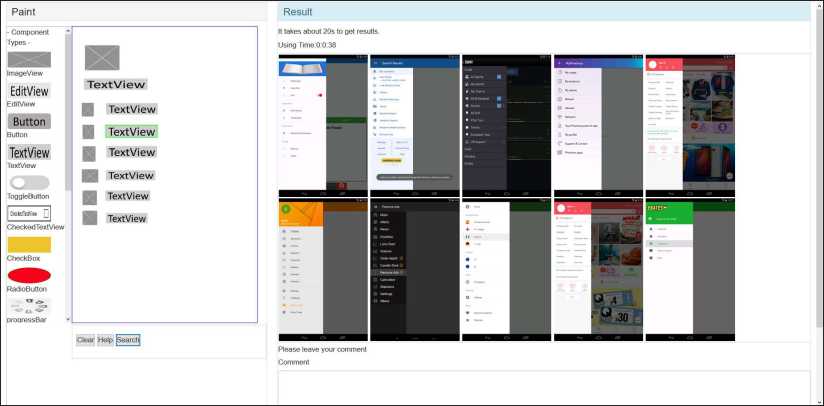


图 8. 我们的用户界面搜索网站的核心页面。

最类似于绘图画布上的线框。用户可以反复细化线框并搜索相关的 UI 设计。

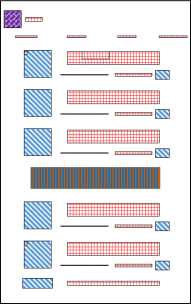
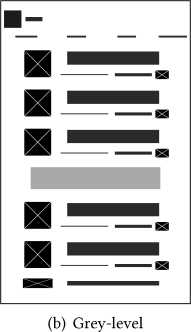
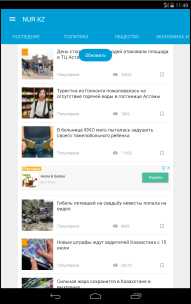
1. 实验设计

在本节中，我们将描述我们在实验中研究的研究问题 (RQ)，然后详细说明每个 RQ 的设置，包括实验数据集、基线模型、指标和程序。

5.1 研究问题 (RQ)

我们的评估旨在回答以下 RQ：

* **RQ1：线框的不同表示的有效性：用于表示线框的**哪种色**觉**表现最好？为什么表现不一样？
* **RQ2：我们的UI 搜索引擎人工数据集的准确度表现：**面对UI 设计的巨大变化，我们的方法在多大程度上实现了找到相关UI 设计的目标？它与基于图像相似性、基于组件匹配或基于朴素神经网络的 UI 设计搜索相比如何，其原因是什么？
* **RQ3：对现实世界用户界面的泛化能力：**从开发人员的角度来看，我们的模型表现如何？它与 RQ2 中的最佳基线相比如何？
* **RQ4：用户对我们工具在实用性和多样性方面的建议的研究：**我们的搜索引擎真的帮助开发人员设计用户界面吗？我们的搜索引擎在推荐 UI 的实用性和多样性方面对他们有多大帮助？



(a) 原件

(c) 色阶

(d) 纹理级

图 9. 不同输入格式的示例。从左到右分别是原始UI图像、颜色级线框、灰度级线框和纹理级线框

5.2 RQ1：线框不同表示的有效性

我们介绍了用于评估不同类型线框有效性的训练数据集和实验数据集，然后是该评估中使用的指标。请注意，用于生成这些数据集的方法和本评估中使用的指标也用于回答 RQ2。

5.2.1*实验数据集。*为了回答 RQ1，我们需要首先使用不同的线框表示作为训练数据集构建几个线框数据集。然后，我们构建了另一个实验数据集，以自动评估由这些不同类型的线框表示训练的三个模型的有效性。

我们研究了三种类型的视觉组件表示，包括不同的灰度值、不同的颜色和具有不同纹理的不同颜色。我们将它们分别表示为灰度级、颜色级和纹理级线框。这三种表示的示例可以在图9 中看到。对于三种训练数据集，我们可以直接使用第[3.1.2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark23)节所述的方法生成它们[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark23)

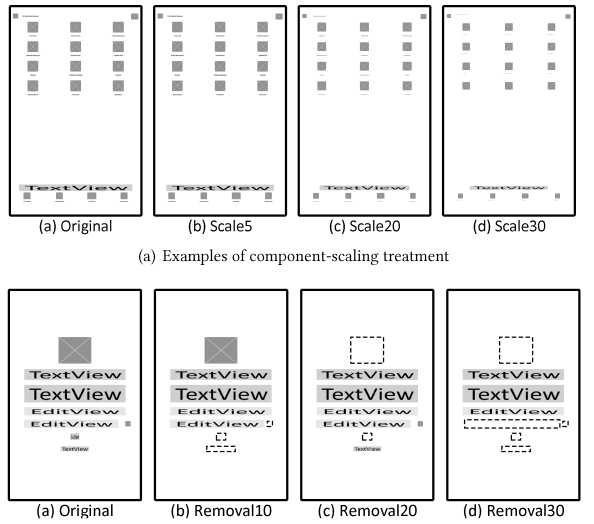
其次，为了根据线框的不同表示来评估 UI 设计搜索方法的性能，我们需要相关 UI 设计的数据集。不幸的是，不存在这样的数据集。也无法在大型 UI 设计数据库中手动注释这样的数据集，以便在面对不同的 UI 设计变化时进行方法能力的大规模实验。受用于增强深度学习模型训练的数据增强方法的启发 [ 34 , 64 ]，我们更改了 Android UI 设计数据库中的 UI 屏幕截图，以人为地创建相关但不同的 UI 设计对。

根据 UI 截图中组件的位置/大小（参见第[3.1](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM" \l "bookmark19" \o "当前文件)节[）](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark19)，我们执行两种适用于 UI 设计的更改操作：*组件缩放* 和*组件移除。*组件缩放是将 UI 线框中的所有可视组件按其原始大小的 5%、10%、15%、20%、25% 或 30% 像素（向上取整）缩小到其中心点（见图10） (a) ) 模拟组件位置/尺寸的设计变化。组件去除是随机去除一些覆盖所有组件总面积10% ± 5%、20% ± 5%或30% ± 5%的视觉组件（见图10（b））) 模拟组件类型/数量的设计变化。为简单起见，我们将组件去除处理中的10% ± 5%、20% ± 5% 或 30% ± 5% 表示为去除 10、去除 20 和去除30 。如图2 中的示例所示，此类设计变化通常是

出现在相关的 UI 设计中。实际上，这两种类型的设计变化可能同时发生。但是我们分别执行这两种类型的更改以研究搜索方法处理不同类型设计变化的能力。

我们从 25 个类别中随机选择两组截图，每组由 500 个截图组成。请注意，对于每个集合，从每个类别中截取的屏幕截图的比例与每个类别在总数据库中的原始比例相同。然后，我们将六种缩放处理应用于第一组，并将三种成分去除处理应用于第二组。第二组 UI 截图应该至少有 5 个 UI 组件，以便移除组件后还剩下一些组件。结果，我们获得了 4500 对原始处理的 UI，它们被认为是相关但不同的 UI 设计。我们有九个实验（每个处理一个）。例如，Scale10 实验使用 10% 组件缩放获得的 UI 作为查询。请注意，我们使用上述三种线框表示形式三次生成了相应的实验线框数据集，这意味着我们有三个实验数据集，每个数据集包含 4500 对原始处理的 UI。使用此数据集，我们评估搜索方法可以使用处理过的 UI 设计作为查询，根据线框的不同表示检索数据库中原始 UI 的效果。

5.2.2*评估指标。*我们通过两个指标评估 UI 设计搜索方法的性能：Precision@k (Pre@k) (k=1) 和 Mean Reciprocal Rank (MRR)。度量值越高，搜索方法的性能就越好。Precision@k 是相关 UI 设计的查询 UI 的 top-k 结果的比例。由于我们将原始 UI 视为本研究中处理过的 UI 的唯一相关 UI，因此我们使用最严格的指标 Pre@1：Pre@1=1 如果第一个返回的 UI 是原始 UI，否则 Pre@1=0。MRR 计算所有查询 UI 的搜索结果中第一个相关 UI 设计的倒数排名（即 1/r）的平均值。



(b) 成分去除处理的例子

图 10. 组件缩放和组件去除处理示例

* 1. RQ2：我们的 UI 搜索引擎的准确度表现

在这个 RQ 中，我们使用了 RQ1 中探索过的线框的最佳表示。我们评估我们的方法在面对巨大的 UI 变化时如何实现找到相关 UI 设计的目标，以及它与基于图像相似性、基于组件匹配或基于朴素神经网络的 UI 设计搜索的比较，以及什么是这背后的原因。在下文中，我们将介绍此 RQ 中使用的实验数据集和指标，然后详细说明所使用的基线模型。

*5.3.1 数据集和指标。*为了自动评估和比较我们的模型和基线的有效性，我们使用相同的处理方法来构建实验数据集，而此时我们只需要考虑一种线框表示。我们再次从与 RQ1 中的数据不同的 25 个类别中随机选择 500 张 UI 图像来构建实验数据集，然后应用第[5.2.1](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark48)节所述的九种处理方法[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark48)我们总共有 4500 对原始处理的 UI，它们被认为是相关但不同的 UI 设计。使用此数据集，我们评估搜索方法可以使用处理过的 UI 设计作为查询在数据库中检索原始 UI 的效果。此外，我们还采用了第[5.2.2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark49)节中所述的相同指标

*5.3.2 基线方法。*我们考虑了四个基线：其中两个分别使用简单的颜色直方图和高级 SIFT 特征计算图像相似度，第三个实现了 GUIFetch [ 19 ]提出的组件匹配启发式，最后一个使用具有完全连接层的朴素神经网络来自 Rico [ 35 ]。这四个基线用于评估 RQ2，RQ2 的结果将决定 RQ3 中使用的基线。

**基于图像特征的相似性。**颜色直方图是一种简单的图像特征，表示每个 RGB（红、绿、蓝）通道中颜色的分布。它已广泛用于图像索引和图像检索 [ 40 , 41 , 45 ]。尺度不变特征变换 (SIFT) [ 54 ] 是一种高级图像特征，广泛用于图像检索 [ 48 , 73 ]、对象识别 [ 54 ]、图像拼接 [ 24]]。它定位图像中的关键点，并利用关键点的局部特征来表示图像。不同的图像可以有不同数量的关键点，但每个关键点都表示在同维特征向量中。这两个基线通过图像特征相似度返回前 k 个最相似的 UI 设计。

**基于启发式的组件匹配。**GUIFetch [ 19 ] 是最近提出的一种技术，用于通过从两个 GUI 之间的匹配组件计算出的相似性度量来搜索相似的 GUI。它匹配相同类型的组件。根据两个组件的x坐标、y坐标、长度和宽度的差异计算两个组件的相似度。如果一个因素的差异在给定的阈值内，则相似度分数增加 10，否则为 0。 在计算两个 UI 中每对组件的相似度分数后，它使用二部匹配算法 [ 47] 以确定最佳组件匹配。将匹配的组件的相似度分数相加，然后除以查询 UI 中的组件可以具有的最大相似度值（即 40 乘以查询 UI 中的组件数）。然后它将具有最高相似度得分的前 k 个 UI 返回到查询 UI。

**基于神经网络的匹配。**Rico [ 35 ] 是一个 UI 数据集，用于支持 UI 设计领域的各种任务。它展示了基于自编码器框架内具有六个完全连接层的朴素神经网络的 UI 搜索的潜在用途。来自他们模型的潜在向量被用作他们的线框数据集的特征。在推理方面，他们首先提取查询线框的潜在向量，将其与他们数据集的潜在向量进行比较，然后返回最近的邻居作为推荐。为了公平比较，我们采用他们论文中提到的相同配置在我们的数据集上训练模型。

* 1. RQ3：对现实世界用户界面的泛化能力

为了在应用于现实世界的应用程序时进一步评估我们的模型，我们对 UI 设计搜索结果的相关性进行了人工评估。我们通过使用未见过的 UI 设计作为查询来搜索 UI 设计数据库来确认我们模型的泛化并回答 RQ3。为此，我们需要构建另一个看不见的 Uls 数据集。我们将在下面介绍这个数据集、人工评估程序和指标。请注意，此 RQ 中的基线由 RQ2 的结果定义。我们只考虑来自 RQ2 的几个（不是全部）基线，因为对 UI 设计搜索结果相关性的人工评估是劳动密集型的。

*5.4.1 未见查询 Uls 的数据集。*我们工具的 UI 设计数据库包含来自 25 类 Android 应用程序的 UI 屏幕截图。我们为每个类别随机下载一个尚未包含在我们的概念验证实施中的应用程序。使用相同的逆向工程方法 [ 27 ] 来获取这个新下载的应用程序的 UI 屏幕截图。我们为收集的 UI 屏幕截图生成相应的 UI 线框，如第[3.1](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark19)节所述[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark19)我们为每个应用程序选择两个 UI 线框，并获得 50 个 UI 线框作为本研究中的查询 UI 设计。根据我们的观察，选定的 UI 线框包含视觉组件的变体类型和数量。

*5.4.2 UI 设计相关性的人工评估。*我们从我们学校招募了五名参与者，P1、P2、P3、P4 和 P5 作为人类注释者。他们已经从事 Android 应用程序开发至少两年了。对于每个查询 UI，我们分别通过我们的方法和 GUIFetch 基线获取前 10 个搜索结果（即总共 20 个 UI 设计）。为避免预期偏差，这 20 个 UI 设计随机混合在一起，以便人工注释者不知道通过哪种方法返回哪个 UI 设计以及该 UI 设计在搜索结果中的排名。两个注释者独立检查 UI 设计搜索结果。对于每个查询 UI 线框，他们将 20 个 UI 设计中的每一个分类为与查询 UI 相关或不相关。注释者将获得查询 UI 线框的原始 UI 屏幕截图作为比较参考。

*5.4.3 评估指标。*我们使用两种统计方法来衡量两个人类注释者之间以及所有五个人类注释者之间的评价者间一致性。对于第一个指标，我们计算 Cohen 的 kappa 统计量 [ 33 ]，该统计量适用于衡量两个评估者访问两个类别的多个项目之间的一致性。对于第二个指标，我们计算 Fleiss 的 kappa 统计量 [ 38]，用于评估多个评估者之间的一致性。基于五个注释者对 UI 设计相关性的判断，我们通过三种策略将返回的 UI 设计视为相关：严格（两个注释者都将其标记为相关）中等（大多数注释者将其标记为相关）和宽松（至少一个注释器将其标记为相关）。然后我们计算 Precision@k (k=1, 5,10) 和 MRR。我们在本研究中不使用召回率和平均平均精度 (MAP)，因为不可能在大型 UI 设计数据库中手动注释查询 UI 的所有相关 UI 设计（我们的概念验证实现中的 54,987 个 UI 屏幕截图） .

* 1. RQ4：用户对来自我们工具的建议在实用性和多样性方面的研究

为了回答 RQ4 并评估我们搜索引擎的有用性，我们进行了一项用户研究。我们从 Daily UI design challeng [e [[11]](#footnote-11)[6] 中[[12]](#footnote-12)](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark5)选择了五个 UI 设计任务[，](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark5) 招募 18 名学生来设计这些任务，使用我们的工具搜索相关的 UI 并修改他们的草稿，然后对推荐的有用性和多样性进行评分。下面，我们将介绍这五个任务的详细信息、实验过程和此 RQ 中使用的指标。

*5.5.1 UI 设计任务。*我们从 Daily UI designchalleng [e [[13]](#footnote-13)[7] 中[[14]](#footnote-14)](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark5) 选择了五个 UI 设计任务[：](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark5)*注册、图库、登录、首选项设置*和*导航抽屉。* 这五个 UI 涵盖了 Android 移动应用程序的基本功能：*注册* 和*图片库*分别是用于收集用户输入和显示信息内容的典型 UI。*登录*是用户认证的常用功能，*偏好设置*通常用于软件定制。*导航抽屉*是一项核心交互功能，可让用户访问所有应用程序功能。此外，即使对于作为本研究目标用户的非专业 UI 设计师来说，这些功能也很容易理解

*5.5.2 实验程序。*我们通过学校的邮件列表从我们学校招收18名学生。尽管有 6 名学生具有一定的前端软件开发经验，但没有一个参与者具有 Android UI 设计经验。换句话说，他们是没有经验的设计师，与第[2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark13)节中的露西相同[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM" \l "bookmark13" \o "当前文件)参与者被赋予五个 UI 设计任务，并被要求为每个任务设计一个 UI 线框。每个任务分配 15-30 分钟。由于时间限制，我们不要求参与者设计 UI 的高保真视觉效果。为了协助他们的设计工作，参与者使用我们的网络工具绘制 UI 线框并搜索我们的 54,987 个 Android UI 设计数据库（参见第[4](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark31)节[）](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark31). 该工具返回查询 UI 的前 10 个 UI 设计。我们为参与者提供工具使用教程和 15 分钟的热身课程，以学习使用该工具。对于每项任务，参与者可以根据需要进行多次搜索。对于最后一次搜索，他们被要求在搜索结果中选择他们认为与他们绘制的查询 UI 线框相关的 UI 设计。他们还被要求通过 5 点李克特量表（1 为最低，5 为最高）对上次搜索的搜索结果的整体多样性和有用性进行评分。详细而言，有用性是指当参与者面临真正的 UI 设计任务时，有用的搜索结果如何帮助他们理解/调整设计选项。例如，当参与者正在搜索一些与构建注册页面相关的 UI 时，我们搜索引擎的建议符合他们的设计要求。多样性是指推荐结果的多样性，例如，推荐的UI是否涉及可能超出预期的可变组件使用/布局或颜色/大小/字体效果。

*5.5.3 评估指标。*我们记录参与者对每个任务的搜索次数。基于上次搜索的 UI 设计搜索结果的相关性判断，我们计算 Precision@k (k=1, 5,10) 和 MRR。我们不报告 Recall 和 MAP，因为不可能在我们的数据库中为用户绘制的 UI 线框图标注所有相关的 UI 设计，该数据库包含 54,987 个 UI 屏幕截图。

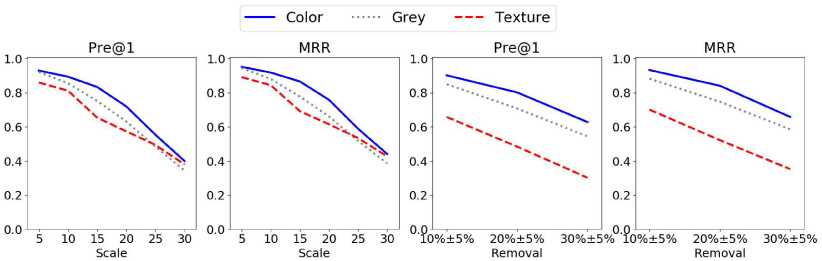
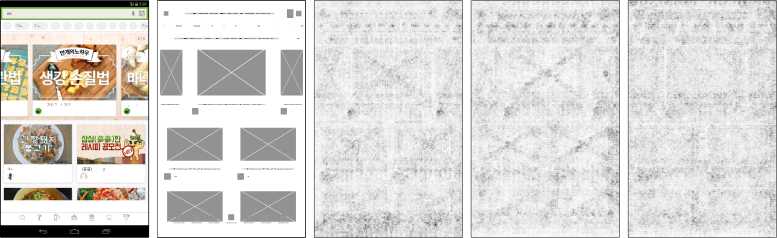


图 11. 三种表示的结果



(a) 原始截图 (b) 线框 (c) 灰度级 (d) 颜色级 (e) 纹理级

图 12. 灰度级、颜色级和纹理级线框的热图

颜色级别模型和纹理级别模型的性能，特别是在组件去除处理中。这可能是因为纹理信息太复杂，可能会在几个 max-pooling 层之后混淆模型。

*6.1.2 CNN 可视化。*为了更好地理解不同表示的影响，我们使用图[12 中的普通](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark58)（即标准）反向传播显着性 [ 25 ]将这些模型可视化[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark58)我们可以发现颜色级别模型的热图最清晰，而纹理级别模型的热图最模糊，噪点较多。灰度热图比颜色一级更模糊，因为灰度一级的组件和背景之间的差异很小。总之，颜色级别模型表现最好，我们选择它作为我们线框数据集的表示。

6.2 RQ2 结果

我们基于深度学习的 UI 设计搜索方法是同类技术中的首创。它旨在在面对 UI 设计的巨大变化时找到相关的 UI 设计。在此 RQ2 中，我们的目标是评估我们的方法实现此设计目标的程度，以及它与基于图像相似性或基于组件匹配的 UI 设计搜索的比较情况。我们在本次评估中使用颜色级线框数据集，因为这种表示的有效性已在第[6.1](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark7)节中得到证明[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark7)

*6.2.1 运行时性能。*我们在配备 Intel i7-7800X CPU、64G RAM 和 NVIDIA GeForce GTX 1080 Ti GPU 的机器上运行实验。取Scale10实验的推理时间

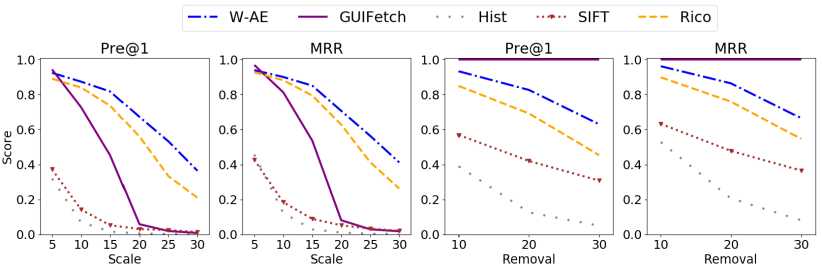


图 13. 九个自动实验的结果

举个例子。我们的 W-AE（Wireframe Autoencoder 的缩写）、Rico、GUIFetch、SIFT 和 colorhistogram 分别需要 561.2 秒、771.4 秒、7446.9 秒、3944.6 秒和 523.6 秒进行 500 次查询。一般来说，W-AE 比 GUIFetch 和 SIFT 基线快 12 倍和 6 倍，并且与颜色直方图和 Rico 基线一样快。

*6.2.2 检索性能。*图13显示了九个将 UI 作为查询处理的实验中五种方法的性能指标。颜色直方图基线和 SIFT 基线在所有组件缩放实验中都具有相近的性能。在组件缩放比例为 10% 时，它们的性能指标低于 0.2，在比例为 20% 或更高时，它们的性能指标接近于 0。 对于组件去除实验，高级 SIFT 特征比简单的颜色直方图特征表现更好. 这是因为移除组件处理的 UI 仍然有许多完整的组件（见图10(b))，它们与原始 UI 中的对应点具有相同的关键点（因此具有相同的 SIFT 特征）。这有助于为经过组件移除处理的 UI 检索原始 UI。然而，在组件去除率 20% 或更高时，SIFT 基线的性能指标低于 0.5。

相比之下，我们的 W-AE 在面对较大的组件缩放和组件移除变化时更加稳健，因为我们的 CNN 模型可以通过深度神经网络提取更抽象、复杂的 UI 设计相关图像特征，而这些特征对图像的敏感度要低得多。与颜色直方图或 SIFT 等低级图像特征的差异。在组件缩放比例为 20% 时，我们的 W-AE 仍然实现了 70.0% 的 Precision@1 和 0.73 MRR。当组件缩放比例为 25% 或更高时，我们的 W-AE 的性能会下降（但仍然比四个基线好得多）。这是因为许多小尺寸的视觉组件（例如复选框、开关或小文本）即使是人眼也几乎看不到（见图10（a））。同样，这些极小的组件的特征将变得对 CNN 模型的“眼睛”（即卷积核）不可见，因此无法用于 UI 设计相似性的测量。虽然这种极小的 UI 组件可以在极端条件下测试搜索方法的极限，但它们很少存在于真实的 UI 中，因为它们对用户不友好。在 20% 的组件去除率下，我们的 W-AE 仍然实现了 84.6% 的 Precision@1 和 0.88 MRR。在组件去除率为 30% 时，模型性能下降（仍然比基线好得多）。但是，如图10(b) 中的示例显示，组件覆盖面积比原始 UI 少 30% 的处理过的 UI 可能变得不再与原始 UI 相似。但是在我们的自动实验中，我们仍然将原始 UI 视为处理过的 UI 的基本事实，这可能会导致所有评估方法的度量有偏差。

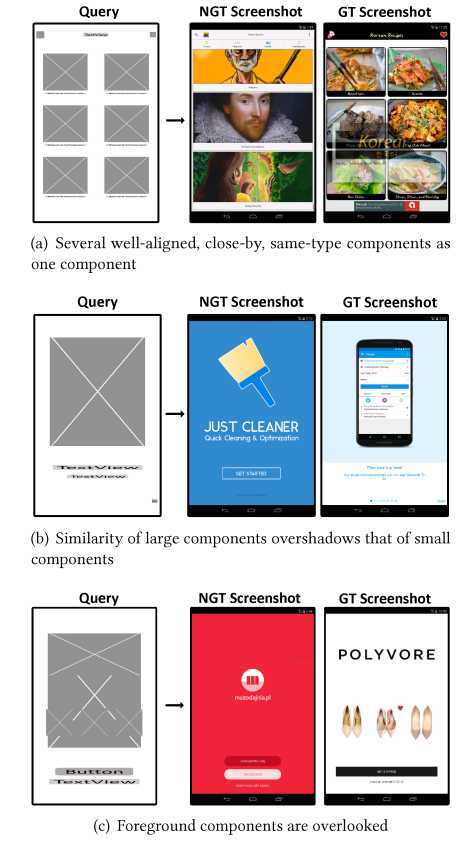


图 14. 排名在 ground-truth (GT) UI 之前的 non-ground-truth (NGT) UI 示例

我们的 W-AE 在所有指标上都大大优于 Rico。Rico 基线比其他基线表现更好，但随着 UI 组件和布局相似性的降低，Rico 和我们的 W-AE 模型之间的性能差距不断扩大。在组件缩放实验中，Rico 基线在缩放比例为 5%-10% 时与我们的 W-AE 相当，但在比例为 25% 或更高时会迅速退化。这是因为 Rico 应用了全连接层，它考虑了 UI 中的每个像素，而不会过滤掉无意义和嘈杂的像素，从而导致对小的输入变化敏感。相比之下，我们的 W-AE 执行卷积和池化策略以从更稳定的 UI 中提取核心特征。类似的观察也适用于去除处理的实验。ACM 翻译。软件 英。方法论，卷。29、3、19条。发表日期：2020年5月。

在比率为 10% 时性能良好，然后在比率为 20% 时下降到 70.6% Precision@1 和 0.77 MRR，在比率为 30% 时下降到 47.6% Precision@1 和 0.57 MRR。

在所有组件缩放实验中，GUIFetch 基线仅在缩放比例为 5% 时实现了与我们的 W-AE 相当的性能。但是，GUIFetch 基线的性能在组件缩放比例增加时显着下降，并且在缩放比例为 20% 或更高时接近于 0。这是因为 GUIFetch 的组件匹配规则的敏感性（参见第[5.3.2 节](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM" \l "bookmark52" \o "当前文件)[）](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark52) . 当组件缩放比例较大时，处理后的 UI 和原始 UI 中对应组件的位置和大小在阈值下将不再足够接近，从而无法匹配。对于所有组件移除实验，GUIFetch 基线“不出所料”实现了完美的性能（所有指标均为 1.0）。这种完美的性能是因为留在处理过的 UI 中的所有组件都是完整的，因此可以匹配原始 UI 中的对应组件。此外，GUIFetch 的相似性度量仅考虑查询 UI 中匹配与不匹配的组件。因此，处理后的 UI 和原始 UI 的相似度得分为 1.0。然而，

*6.2.3 检索故障分析。*为了更深入地了解我们的 CNN 模型对 UI 设计的视觉语义进行编码的能力，我们手动检查了真实 (GT) UI 排在其他非真实 (NGT) UI 之后的检索失败案例，并确定我们自动实验中检索失败的三个主要原因。图14显示了这三种检索失败的典型示例。

首先，查询 UI 包含几个对齐良好的、靠近的、相同类型的组件，但模型返回一些具有相同类型但在相应 UI 区域中组件数量越来越少的 UI（图14 (a) ）。这揭示了我们的模型在区分几个对齐良好的、靠近的、相同类型的组件方面的局限性。然而，一定程度的建模模糊性对于检索具有不同组件数量的类似 UI 设计很重要（例如图2 (g) 与 (h)/(i)/(j) 以及图16 中的图库示例））。需要注意的是，我们在自动实验中使用成对的原始 UI 和处理过的 UI 作为相关的 UI 设计，并将数据库中的所有其他 UI 视为与查询 UI“无关”。然而，如图14 (a) 中的示例所示，非真实用户界面仍然可以与查询用户界面相关。这种 UI 设计相关性只能由人类来判断，就像我们在泛化实验和用户研究中所做的那样。

其次，查询UI包含一个覆盖UI大面积的UI组件（通常是一个ImageView），模型返回一些与查询UI相似的UI，只有那个大的组件，但UI的其他部分不相似设计（图14 (b)）。这样的检索结果表明我们的模型没有等效地处理来自小或大视觉组件的特征。这种不等价的处理是合理的，因为大组件在视觉上更明显，但它可能导致大组件的相似性掩盖了小组件的相似性。

第三，查询UI包含一些与大背景组件重叠的前景UI组件，但返回的UI仅包含没有前景组件的背景组件，尤其是当前景和背景组件属于同一类型时。重叠的组件，尤其是相同类型的组件，在 UI 线框图中可能在视觉上难以区分，因为它们缺乏高保真视觉效果（例如，不同的颜色或图像）来区分它们。它们对我们基于线框的 UI 设计搜索构成威胁。然而，根据我们的观察，大多数具有重叠组件的 UI 设计都有一个背景图像，其上放置了真正功能性的 UI 组件。通过在生成 UI 线框时删除此类背景图像，可以减轻这种威胁。

表 1. 评估人间协议的成对比较

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **P1** | **P2** | **P3** | **P4** | **P5** |
| **P1** | —— | 0.43 | 0.37 | 0.45 | 0.48 |
| **P2** | 0.43 | —— | 0.38 | 0.51 | 0.49 |
| **P3** | 0.37 | 0.38 | —— | 0.43 | 0.42 |
| **P4** | 0.45 | 0.51 | 0.43 | —— | 0.56 |
| **P5** | 0.48 | 0.49 | 0.42 | 0.56 | —— |

表 2. 人类相关性评估结果

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **轻松** | | **缓和** | | **严格的** | |
| **声光发射** | **图形界面抓取** | **声光发射** | **图形界面抓取** | **声光发射** | **图形界面抓取** |
| **预@1** | **0.84** | 0.64 | **0.5** | 0.32 | 0.14 | **0.16** |
| **预@5** | **0.77** | 0.65 | **0.47** | 0.34 | **0.20** | 0.13 |
| **预@10** | **0.75** | 0.62 | **0.43** | 0.31 | **0.15** | 0.12 |
| **MRR** | **0.90** | 0.78 | **0.62** | 0.48 | **0.27** | 0.24 |

*面对 UI 设计中组件缩放和组件移除的变化，我们基于 CNN 的方法对整个 UI 设计的视觉语义进行建模，明显优于基于图像相似性和基于组件匹配的方法。但是，我们方法的性能可以通过对对齐良好、邻近组件、小尺寸组件和重叠组件进行建模的能力得到进一步增强。*

* 1. RQ3 结果：泛化评估

基于[6.2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark59)节自动评估中三种基线方法的性能结果[，](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark59)我们在本研究中使用 GUIFetch 基线。我们不使用颜色直方图和 SIFT 基线有两个原因。首先，我们的自动评估表明，即使面对人工设计变化，颜色直方图和 SIFT 基线的性能也很差。其次，人工评估 UI 设计搜索结果的相关性是劳动密集型的，考虑多两个基线将使人工评估的工作量增加一倍。

*6.3.1 结果。*所有参与者都花了大约 120 分钟来评估 1000 个 UI 设计与其相应的查询 UI 线框的相关性。表1显示了所有五个参与者之间成对比较的 Cohen's kappa 结果。对于这些比较，大多数 kappa 统计数据都在 0.42-0.56 的范围内，这表明两者之间存在中等至实质性的一致性。我们进一步进行了 Fleiss 的 kappa [ 38] 评估所有评估者之间的一致性。我们的方法和 GUIFetch 基线对 UI 设计搜索结果的 2500 (500x5) 个注释的 Fleiss kappa 分别为 0.41 和 0.51。我们认为这种程度的一致性是可以接受的，因为根据人类注释者的不同背景、经验、教育甚至文化，确定 UI 设计的相关性可能相当主观。

根据我们的观察和采访，这些注解者最关心的有四个方面，包括UI的语义（即功能）、布局、组件的类型和组件的数量。一些参与者更关注某些方面，而另一些参与者则更关注其他部分。一些参与者比较严格，而另一些参与者则相对宽松。与图像分类等手动标记任务不同，有

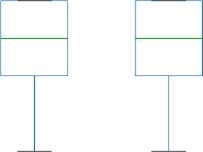
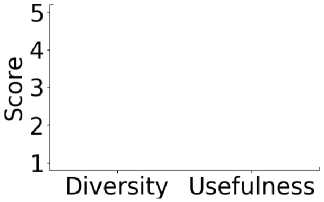


图 15. 参与者对多样性和有用性评级的箱线图

检查每个推荐结果没有严格的正确或错误答案。因此，Cohen 和 Fleiss 的 kappa 率并不高。然而，总而言之，所有参与者的总体反馈定量结果仍然反映我们的方法比我们稍后讨论的基线要好得多。

表[2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark62)显示了我们的方法和 GUIFetch 基线的性能指标。我们的 W-AE 在宽松和中等策略中显着优于 GUIFetch 基线，并且在严格策略中保持优于 GUIFetch 的优势。通过宽松的策略，我们的 W-AE 具有与其在缩放 10% 和去除 20% 实验中的性能相当的性能（见图13）。在中等策略中，我们的 W-AE 仍然达到了 precision@1=0.5 和 MRR=0.62。通过严格的策略，我们的 W-AE 仍然比 GUIFetch 略有优势。由于这项研究涉及许多变体，因此很难就五个参与者达成一致。现实中的 GUIFetch 基线不再像在组件移除实验中那样具有完美的性能。它的性能也比一些 GUIFetch 表现良好的缩放实验差很多。

*我们基于 CNN 的方法可以为一组不同的、看不见的实际应用程序查询 UI 稳健地检索相关的 UI 设计。相比之下，基于单个组件匹配的启发式方法为这些实际应用程序查询 UI 找到的相关 UI 设计要少得多。*

* 1. RQ4 结果：有用性评估

最后，回答 RQ4 并评估我们的 UI 设计搜索引擎在现实世界的 UI 设计任务中的有用性，在这些任务中，用户动态设计和绘制 UI 线框图，在此期间他们使用我们的 UI 设计搜索引擎搜索相关的 UI 设计。

*6.4.1 结果。*18 名参与者在五个 UI 设计任务中总共执行了 168 次搜索。在 90 个参与者任务会话中，59 个有 1 个搜索，12 个有两个搜索，20 个有 3 个或更多搜索。考虑到每个任务的实验时间短，以及绘制 UI 线框的时间，搜索次数是合理的。根据参与者对搜索结果的相关性判断，我们的 W-AE 达到了 precision@1=0.44、precision@5=0.40、precision@10=0.38、MRR=0.59。这些性能指标介于我们生成研究中严格和宽松策略的指标之间，这证明了我们的搜索引擎在支持现实世界 UI 设计任务方面的实用性。

图15显示了 18 位参与者对搜索结果的多样性和有用性评级的箱线图。对于这两个方面，我们模型的结果都获得了 4 分的中位数，其中大多数的分数在 3 到 5 之间，这表明我们的 UI 设计搜索引擎是令人满意的。此外，在他们评分的所有 90 个搜索中，参与者将 57 个（63.3%）搜索的搜索结果多样性评分为 4 或 5，并将搜索结果的有用性评分为 4或 5 表示 51 (56.7%) 次搜索。第[2](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark13)节中说明的激励场景实际上源自我们用户研究中一位参与者的设计工作。我们可以观察搜索结果的多样性和实用性，以激发参与者的注册 UI 设计。图16 分别显示了导航抽屉和图片库设计的另外两个搜索结果示例。对于两个用户绘制的 UI 线框图，我们的工具会返回许多用户注释的相关 UI 设计（以蓝色勾号突出显示）。此外，用户对多样性ACM Trans给出 4 或 5 个评级。和搜索结果的有用性，并对搜索结果提供一些积极的反馈和有用的建议。即使对于不相关的 UI 设计（例如，查询导航抽屉线框的第 4 个 UI），我们模型的推荐仍然有意义，因为该 UI 在视觉上与查询线框相似。对于图库搜索结果，除了与查询UI线框具有几乎相同的UI布局的前3个UI设计外，其他返回的UI设计展示了用于设计图库的多种UI布局。

根据参与者的 1 或 2 多样性和有用性评级，我们的搜索引擎没有为 17 次搜索产生令人满意的搜索结果。通过采访参与者，我们确定了搜索结果不满意的两个主要原因。首先，我们的模型倾向于返回与查询 UI 总体相似的 UI 设计。虽然这样提高了搜索结果的多样性，有利于获得设计灵感，但并不能保证在返回的UI设计中存在用户想要的某些特定的UI组件或特定的组件布局（参见搜索结果反馈）图16 中的图片库）。为了解决这个问题，我们可以考虑更高级的模型，例如变分自编码器 [ 36] 当查询 UI 中的某些用户所需的组件或组件布局未出现在搜索结果中时，这可能会造成更大的损失。

其次，一些参与者抱怨我们的模型有时对 UI 中组件的位置很严格。例如，当用户在首选项设置 UI 的中间区域绘制开关按钮时，我们的工具不会返回相关的 UI 设计。但是当他将切换按钮移动到 UI 的右侧时，我们的工具可以返回许多相关的偏好设置 UI。这个例子实际上表明我们的模型很好地学习了偏好设置 UI 的特性，其中切换按钮通常出现在 UI 的右侧。尽管此建模功能可用于过滤掉不相关的 UI，但它可能会使相关 UI 的搜索对于特定组件布局过于严格。为了放松搜索结果，我们可以使用图像的结构相似性 [ 17 ] 或属性图 [65 ] 支持更摘要 UI 组件布局编码，从而更灵活的 UI 设计搜索。

、‘；来11111111111111111111111111111111

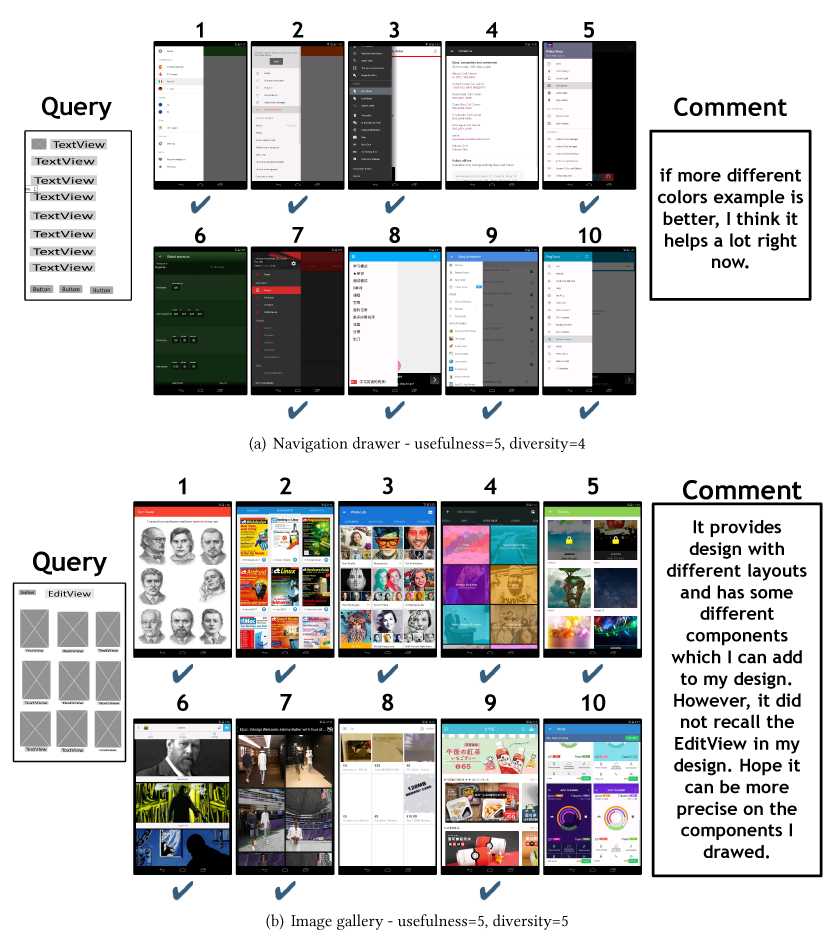


图 16. 我们的用户研究中的搜索结果示例（复选标记表示用户认为设计相关）

1. 对有效性的威胁

我们在工作中讨论了两种类型的有效性威胁，即内部有效性和外部有效性。

* 1. 内部有效性

内部效度是指对因果关系可能影响结果的威胁[ 72]]。首先，我们的自动评估允许我们进行大规模实验以了解我们方法的优缺点，但它只考虑单独的组件缩放和组件移除变化。现实世界的 UI 设计变化会复杂得多。但是，为了深入了解每种处理带来的影响，我们需要控制变量，尝试这两种处理的每一种组合也是不可行的。为了减轻这种影响，我们进一步进行了泛化实验和用户研究，以评估人类参与者的工具。我们的方法在这些研究中的表现与我们的自动评估的表现非常吻合，这让我们对我们的方法在现实世界 UI 设计搜索中的实用性充满信心。

其次，为了确认我们模型的泛化性，我们招募了五名在 Android 开发方面拥有两年以上经验的学生，以手动检查我们模型和基线的结果。然而，相关性概念的概念可能因它们而异，从而影响结果。他们中的一些人可能更强调 UI 的语义含义，而另一些人可能认为由相似组件组成的 UI 是相关的。它们都有意义，因为设计师可以直接从同一场景中重用设计，但也可以从具有相似布局和相似结构的 UI 中获得灵感。保持评价的一致性不同的参与者，我们给了他们一个教程来学习这些概念的一般含义，并有 15 分钟的热身时间来熟悉工具和实验过程。此外，CohenSs kappa 值和 FleissSs kappa 值表明这些参与者之间的一致性中等。这是合理的，因为我们上面提到的变化。我们让五名学生尽可能地避免潜在的偏见，并根据[5.4.3](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark55)节所述的三种策略，即严格、适度和放松的策略来分析结果[。](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark55) 我们断言，通过让五名参与者参与并根据这三种策略分析结果，可以合理减轻这种对有效性的威胁。尽管参与者存在差异，但总体结果仍然表明，我们的方法通过汇总他们的反馈而优于其他基线。

* 1. 外部有效性

外部有效性是指可能限制概括能力的威胁[ 72]。首先，我们的数据收集工具可以从应用程序中收集大部分 UI 元素，但无法捕获 WebView 组件中的详细 HTML 元素和 Uls 中的一些元素，这些元素需要某些特定引擎，例如 unity3d 游戏引擎。因此，这种限制可能会使我们失去 Web 组件和游戏 Uls 的 UI 设计。但是，这些包含 HTML 元素的 UI 中的 GUI 设计应该类似于使用原生元素的那些 UI。不同的实现只是构建用户界面的替代方案，而底层的设计原则应该是相同的。我们收集了来自 7,746 个下载最多的 Android 应用程序的 25 个类别的 54,987 个 UI 屏幕截图的数据库，我们相信这样的大型数据库可以覆盖大多数 UI 设计。我们让我们的工具扩展到收集 WebView 组件和特定引擎中的 UI 元素作为未来的工作。其次，我们的方法是通用的，但我们当前的工具仅支持 Android UI 设计搜索。为了进一步验证我们的方法的通用性，不仅当前的工具应该通过更多来自 Android 应用程序的 UI 设计进一步丰富，而且还应该扩展到其他应用程序（例如，iOS、Web 应用程序）。目前，我们已经在 25 类 Android 应用程序上测试了我们的模型/工具，这在一定程度上展示了我们工具的通用性。由于用户界面的组成在这些平台方面是相似的，我们相信我们的方法也可以通过一些定制来应用。但这需要在未来进一步探索。

1. 相关工作

**UI 设计数据集。**许多UI设计套件[[ 2 ，](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark85)[3 ，](file:///C:/Users/duoduo/Desktop/2021%E5%B9%B411%E6%9C%8828%E6%97%A5000/%E5%A4%8D%E7%8E%B0%E8%AE%BA%E6%96%87.HTM#bookmark86) 8 ，9 ]是在网络上公开。设计师还在社交媒体平台上分享他们的 UI 设计，例如 Dribbble [ 6 ]、UI Movement [ 11]]。它们是设计灵感的重要来源，但它们无法让开发人员接触到真实应用程序的大型 UI 设计空间。此外，这些平台仅支持简单的基于关键字的搜索。或者，现有的应用程序提供了大量的 UI 设计资源库。为了利用这些 UI 设计，人们采用自动 GUI 探索方法来模拟用户与 GUI 的交互并收集现有应用程序的 UI 屏幕截图，这可以支持数据驱动的应用程序，例如 UI 代码生成 [ 27 , 57 , 61 ]、GUI 搜索[ 19 ，22 ，26 ，67 ]，设计采矿[ 51 ]，设计掉毛[ 77]，UI可访问性[ 30 ]，用户交互建模[ 31 ，35 ]和隐私和安全[ 32 ]。同样，我们的工作通过自动 GUI 探索构建了一个大型真实应用程序 UI 设计数据库。与现有工作不同，我们进一步对 UI 截图进行了线化，以支持基于线框的 UI 设计搜索。

**UI 设计搜索。**我们的方法允许用户通过 UI 线框图像搜索 UI 设计。一些技术 [ 22 , 51 , 67 ] 也支持按图像搜索 UI，但它们使用低级图像特征，例如颜色直方图以及其他 UI 信息（如果可用），例如组件类型、显示的文本。其他技术 [ 19 , 66 , 78 ] 支持通过 UI 草图进行 GUI 搜索。但是他们本质上将数据库中的查询 UI 和 UI 都转换为 GUI 组件树，然后找到相似的GUI 通过计算组件树的最佳匹配。与这些作品不同的是，我们的方法真正将 UI 建模为图像，并使用深度学习功能对 UI 的视觉语义进行编码。最相关的工作是 Rico [ 35 ]，它设想了基于深度学习的 UI 搜索的可能性，并展示了几个基于具有高度简化数据的简单全连接层模型的示例。与他们的工作相比，我们开发了一个适用于各种现实生活 UI 设计的复杂模型，实现了一个工作原型并进行了系统的实证研究。

**UI 实现自动化。**Nguyen 和 Csallner [ 61 ] 通过基于规则的图像处理方法检测 UI 屏幕截图中的组件并生成 GUI 代码。它们仅支持一小组最常用的 GUI 组件。更强大的基于深度学习的方法 [ 20 , 27 , 57] 最近有人提出利用自动收集的 UI 屏幕截图和相应代码的大数据。与这些需要高保真 UI 设计图像的 UI 代码生成方法不同，我们的方法只需要 UI 线框，即使对于没有经验的开发人员也可以快速原型化。此外，我们的方法返回一组用于探索设计空间的不同 UI 设计，而不是实现特定 UI 设计的代码。最近的一些作品探索了 UI 设计与其实现之间的问题。莫兰等人。[ 58 ] 通过比较图像与计算机视觉技术的相似性，检查实现的 GUI 是否违反了原始 UI 设计。他们的后续工作 [ 59] 进一步检测和总结不断发展的移动应用程序中的 GUI 变化。UI设计搜索找到相似的UI设计，然后可以应用这些技术来进一步检测相似UI设计之间的差异，这可以帮助细化搜索结果。

1. 结论

本文提出了一种新颖的基于深度学习的 UI 设计搜索方法。我们方法的核心是 UI 线框图像自动编码器。采用图像自动编码器架构消除了障碍，即标记相关的 UI 设计，这是不可能大规模准备的，用于训练 UI 设计编码器。使用从现有应用程序自动收集的大型未标记 UI 线框数据库进行训练，我们的线框编码器学习编码整个 UI 设计的更抽象和更丰富的视觉语义，而不是关键字、低级图像特征和组件类型/位置/大小匹配启发式，导致性能优于基于这些类型的原始信息的搜索方法。我们的方法展示了在支持开发人员探索和了解大型 UI 设计空间方面很有前景的有用性。

致谢

这项研究得到了澳大利亚国立大学 - Data 61 合作研究项目 (CO19314)、澳大利亚研究委员会发现早期职业研究员奖 (DECRA) 资助计划 (DE200100021)、ARC 发现项目计划 (DP170101932) 和桂冠奖学金 (FL00350) 的部分支持）。

1. 1由于谷歌发挥我们的数据收集后，他们的应用程序更新类别，类别（25）我们的数据集的数量是d在谷歌播放当前数（35）ifferent。 [↑](#footnote-ref-1)
2. 所有UI设计图片以及所有实验数据和结果均可在我们的Github存储库<https://github.com/chenjshnn/WAE>下载 [↑](#footnote-ref-2)
3. [↑](#footnote-ref-3)
4. [↑](#footnote-ref-4)
5. <https://github.com/chenjshnn/WAE> [↑](#footnote-ref-5)
6. [↑](#footnote-ref-6)
7. <https://github.com/chenjshnn/WAE> [↑](#footnote-ref-7)
8. [↑](#footnote-ref-8)
9. 我们不会让 K 太大，因为开发人员往往不会浏览一长串推荐 [28,53,68,74,75]。 [↑](#footnote-ref-9)
10. [↑](#footnote-ref-10)
11. <https://www.dailyui.co/> [↑](#footnote-ref-11)
12. [↑](#footnote-ref-12)
13. **实验结果**

    **6.1 RQ1 结果**

    *6.1.1 定量结果。*图11显示了评估三种不同表示线框的结果。总体而言，颜色级别模型在组件缩放和组件移除处理方面仍然比灰度模型略有优势，Pre@1 和 MRR 分别增加了 5%-10% 和 0.01-0.1。原因可能是颜色级模型主要关注线框中包含组件的边界而不是精确的像素值，并且颜色级线框输入包括三个编码更多信息的通道，而灰度级线框仅包括一个通道渠道。相比之下，两者之间的差距很大 [↑](#footnote-ref-13)
14. [↑](#footnote-ref-14)