SEMERU

Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps

Denys Poshyvanyk
W&M

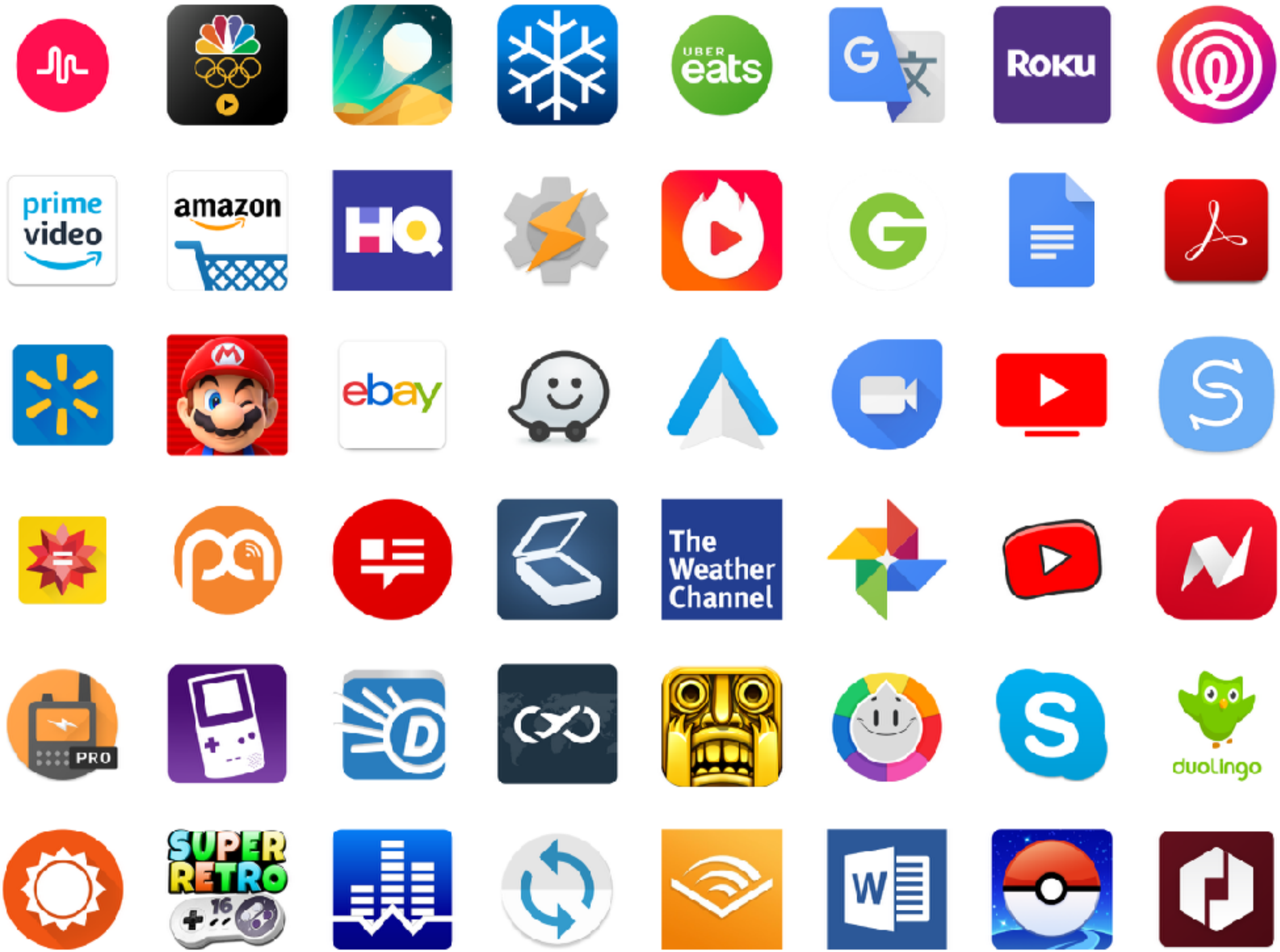
Tue, June 12th, 2018

WILLIAM & MARY
CHARTERED 1693

86% 11:25
11:25 AM
57°F NEW YORK · 04/12
Google
Inbox
Pocket Beta
FeedlyReader
Swarm
Falcon Pro
Calendar
Slack
Play Store

85% 1:14
Google
NOV 15, WED
CLOUDY
53°
Google
Play Store

The App Economy


UBER
eats
Roku
prime
video
amazon
HQ
ebay
The
Weather
Channel
PRO
duolingo
SUPER
RETRO
16
W

Mobile Development Challenges

Highly Competitive App Marketplaces

Noisy Feedback from User Reviews

Change/Fault Prone APIs

Fragmentation

Touch-Based Event-Driven Interface

Android API instability [FSE'13] [TSE'15]

Performance bottlenecks [ICSME'15][MSR'16]

User reviews [ICSME'15]

Energy bottlenecks [MSR'14]

Understanding how developers test apps [ICSME'17]

Mining Android Software Repositories

Crowdsource-based requirements [ICSME'15]

GUI-based testing [MSR'15]

Bug reporting [FSE'15]

Optimizing energy [FSE'15]

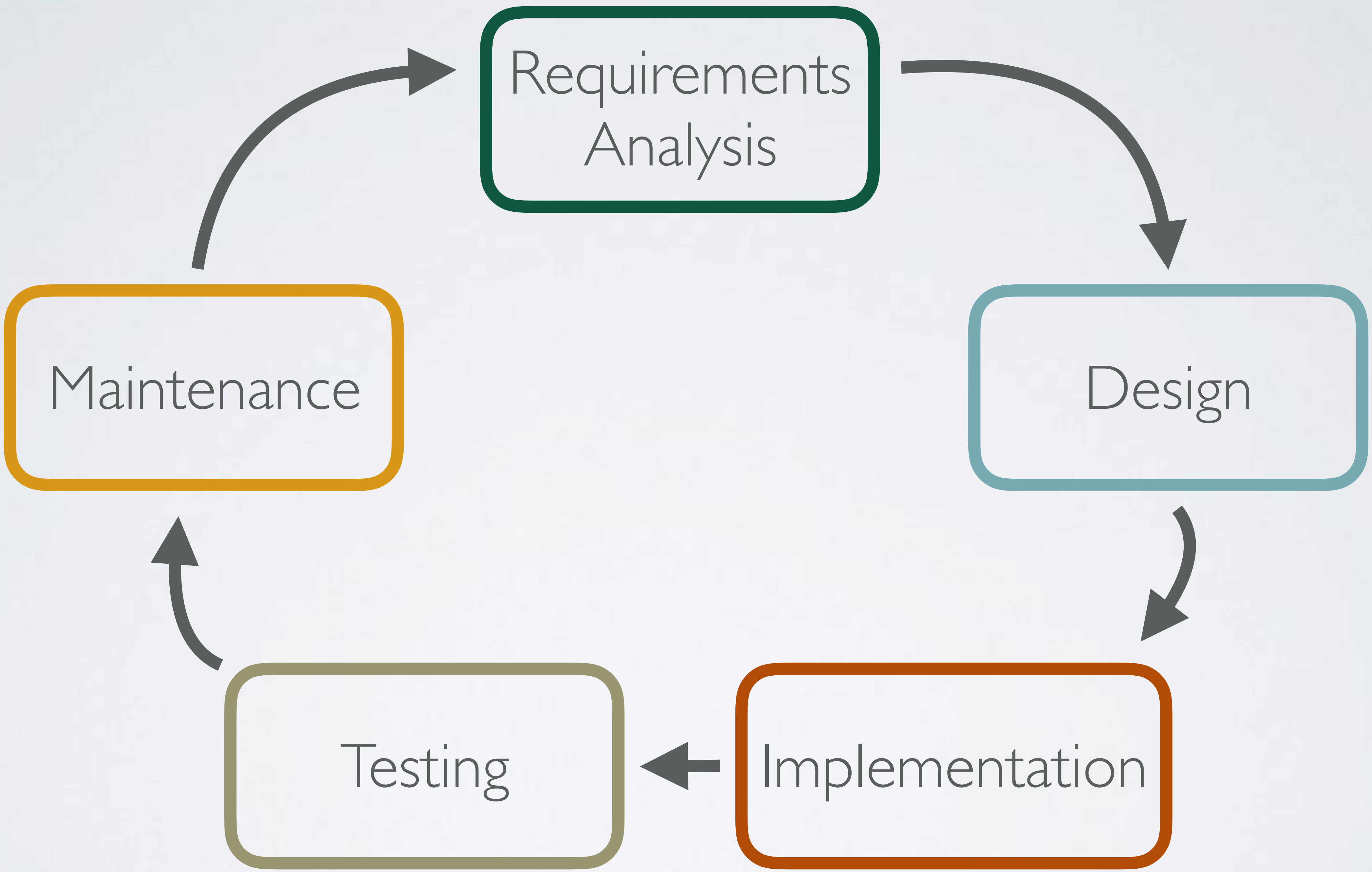
Crash Detection [ICST'16]

Mutation Testing [FSE'18]

Design violations [ICSE'18]

Supporting evolution and maintenance of Android apps

Mobile Development Cycle

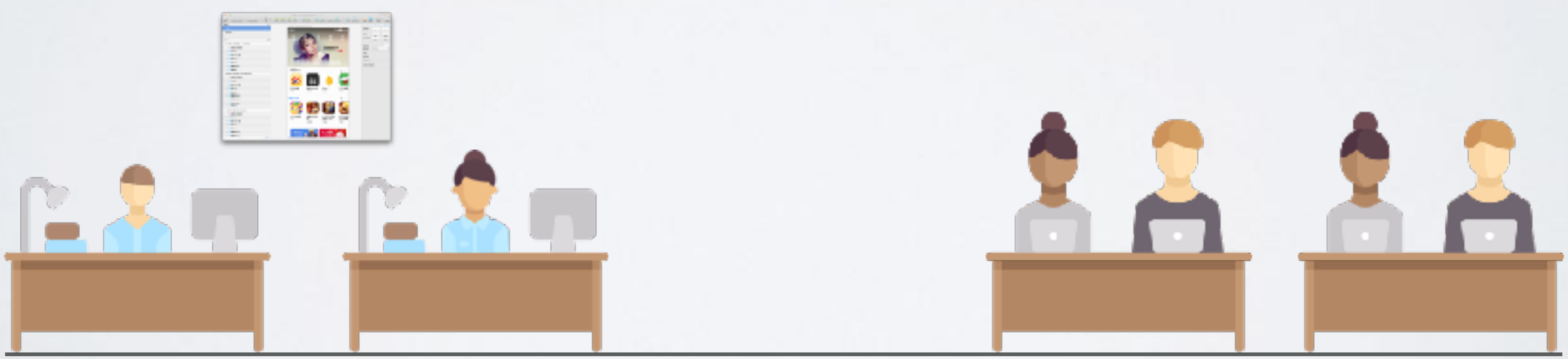


Automated Reporting of GUI Design Violations for Mobile Apps

Research Problem

HUAWEI

Sketch



UI/UX Design Team

Development Team

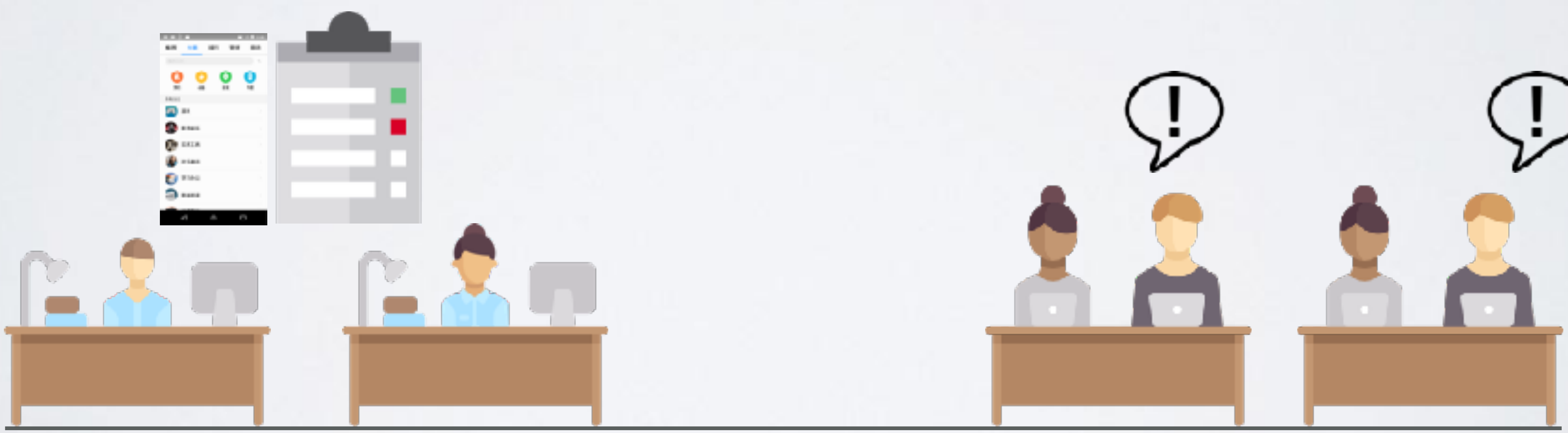
RESEARCH PROBLEM

UI/UX Design Team

Development Team

Research Problem

HUAWEI



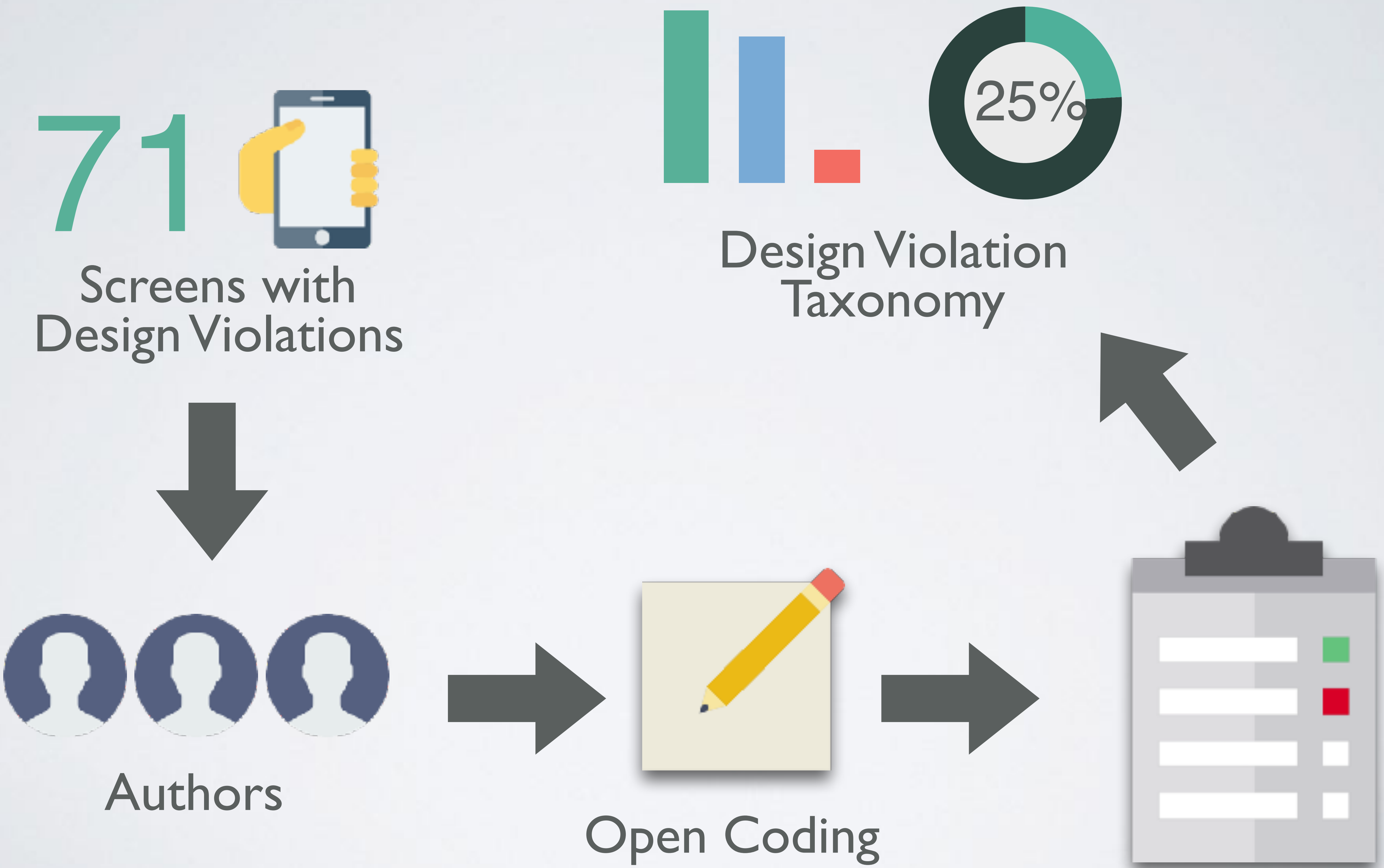
UI/UX Design Team

Development Team

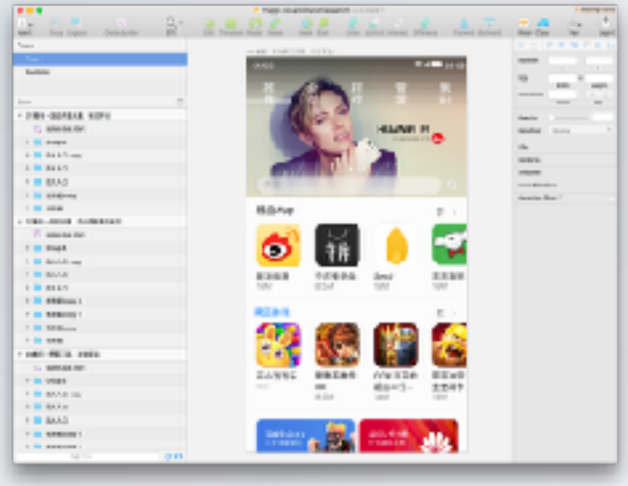
Research Problem

HUAWEI

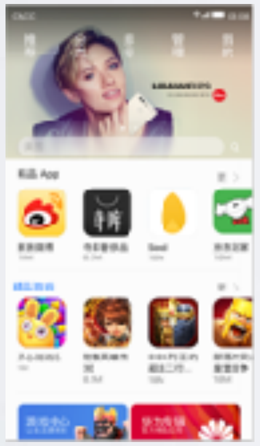
Preliminary Industrial Empirical Study



Our Solution



GUI Design Specifications



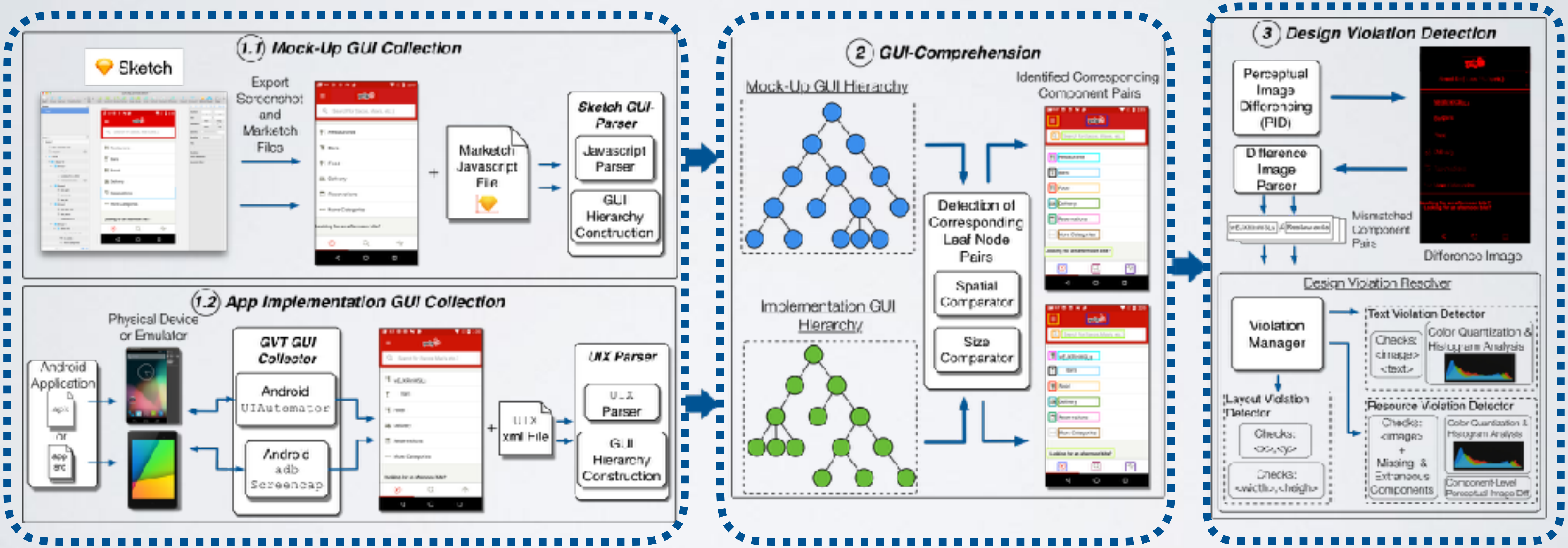
GUI Implementation

GUI VERIFICATION SYSTEM (GVT)

1) GUI-Collection

2) GUI Comprehension

3) Design Violation Detection



The paper appeared at ICSE'18

ICSE 40TH INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING

MAY 27 - JUNE 3 2018
GOTHENBURG, SWEDEN

Automated Reporting of GUI Design Violations for Mobile Apps

Kevin Moran, Boyang Li, Carlos Bernal-Cárdenas, Dan Jelf, and Denys Poshyvanyk
College of William & Mary
Department of Computer Science
Williamsburg, VA, USA
{kpmoran,boyang,cebernal,dkjelf,denys}@cs.wm.edu

ABSTRACT

The inception of a mobile app often takes form of a mock-up of the Graphical User Interface (GUI), represented as a static image delineating the proper layout and style of GUI widgets that satisfy requirements. Following this initial mock-up, the design artifacts are then handed off to developers whose goal is to accurately implement these GUIs and the desired functionality in code. Given the sizable abstraction gap between mock-ups and code, developers often introduce mistakes related to the GUI that can negatively impact an app's success in highly competitive marketplaces. Moreover, such mistakes are common in the evolutionary context of rapidly changing apps. This leads to the time-consuming and laborious task of design teams verifying that each screen of an app was

1 INTRODUCTION

Intuitive, elegant graphical user interfaces (GUIs) embodying effective user experience (UX) and user interface (UI) design principles are essential to the success of mobile apps. In fact, one may argue that these design principles are largely responsible for launching the modern mobile platforms that have become so popular today. Apple Inc's launch of the iPhone in 2007 revolutionized the mobile handset industry (heavily influencing derivative platforms including Android) and largely centered on an elegant, well-thought out UX experience, putting multitouch gestures and a natural GUI at the forefront of the platform experience. A decade later, the most successful mobile apps on today's highly competitive app stores (*e.g.*, Google Play[5] and Apple's App Store[3]) are those that em-

Research Problem

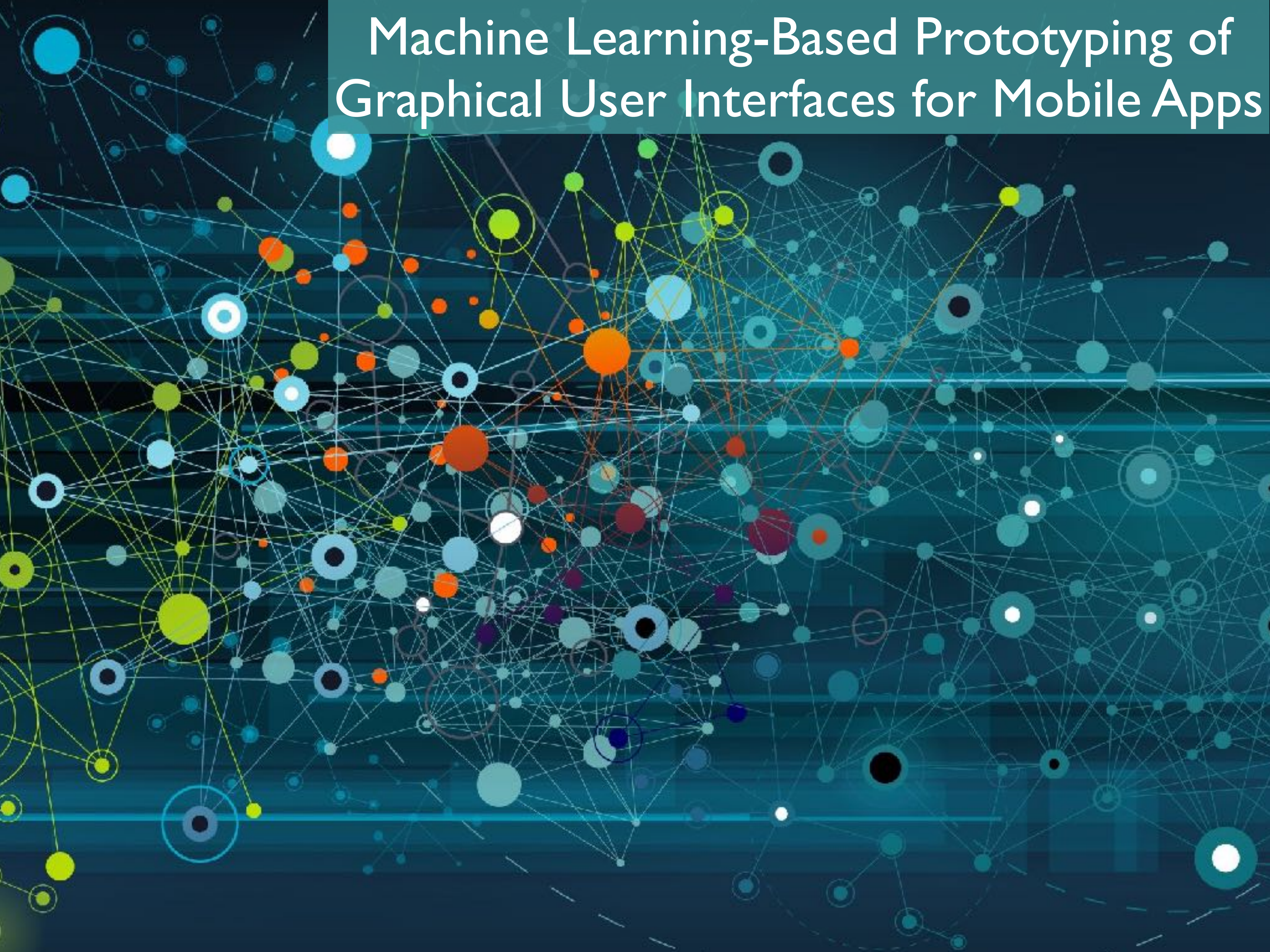
Sketch

Prototype GUI Code

UI/UX Design Team

Development Team

Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps



Existing Work

Reverse Engineering Mobile App Interfaces*

- Utilizes a combination of unsupervised Computer Vision Techniques to Detect Components
- Binary Component Classification (Text or Image)
- GUI-hierarchy generated using heuristics

*Tuan Anh Nguyen and Christoph Csallner. 2015. Reverse Engineering Mobile Application User Interfaces with REMAUI, In Proceedings of the 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE '15)

Neural Machine Translation of Images to Code✦

- Utilizes a Deep-Learning Machine Translation Approach to convert Images into a DSL and then apps
- Only tested on a small synthetic set of apps
- Requires a DSL that must be maintained

✦Tony Beltramelli, "pix2code: Generating code from a graphical user interface screenshot," CoRR, vol. abs/1705.07962, 2017. [Online]. Available: http://arxiv.org/abs/1705.07962

Existing Work

Commercial Solutions

First Principles

How does a developer translate a GUI mock-up into code?

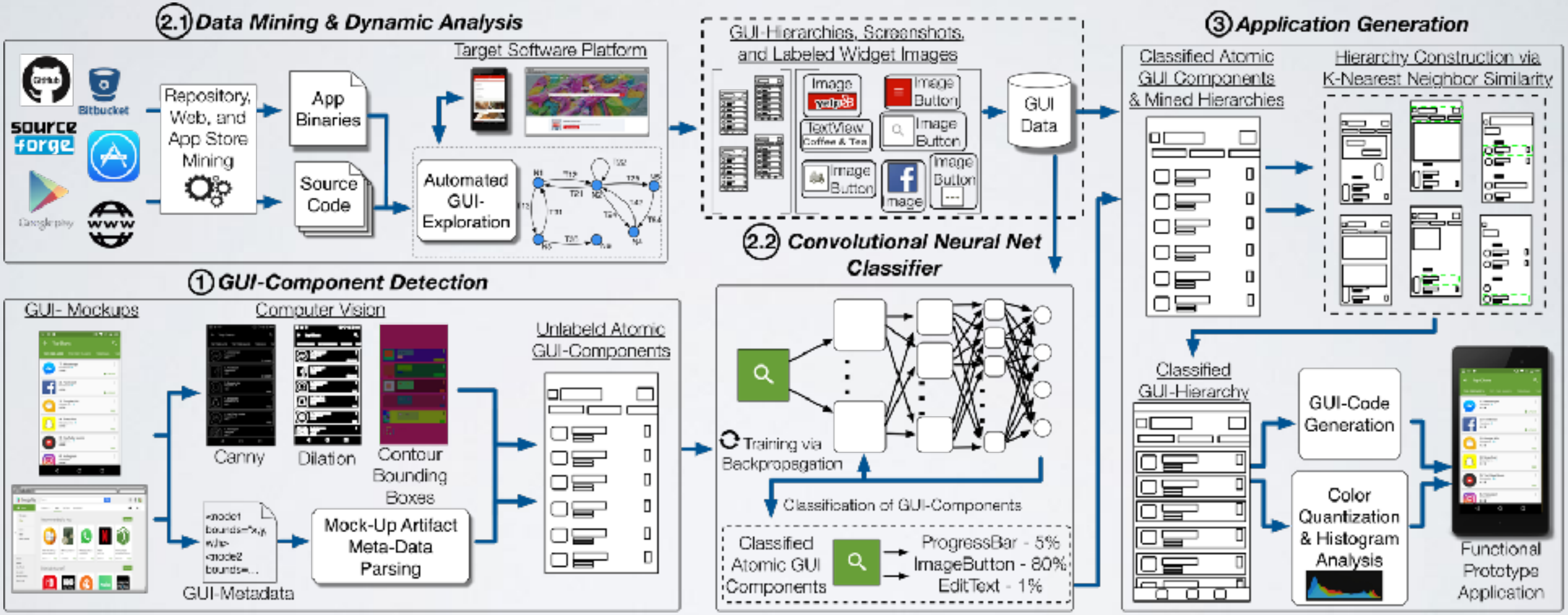
1. ***Detect*** or identify GUI-components that exist in a mock-up

Detection — Classification — Assembly

2. ***Classify*** these GUI-components into their constituent types

3. ***Assemble*** the GUI-components into a hierarchy and stipulate styles

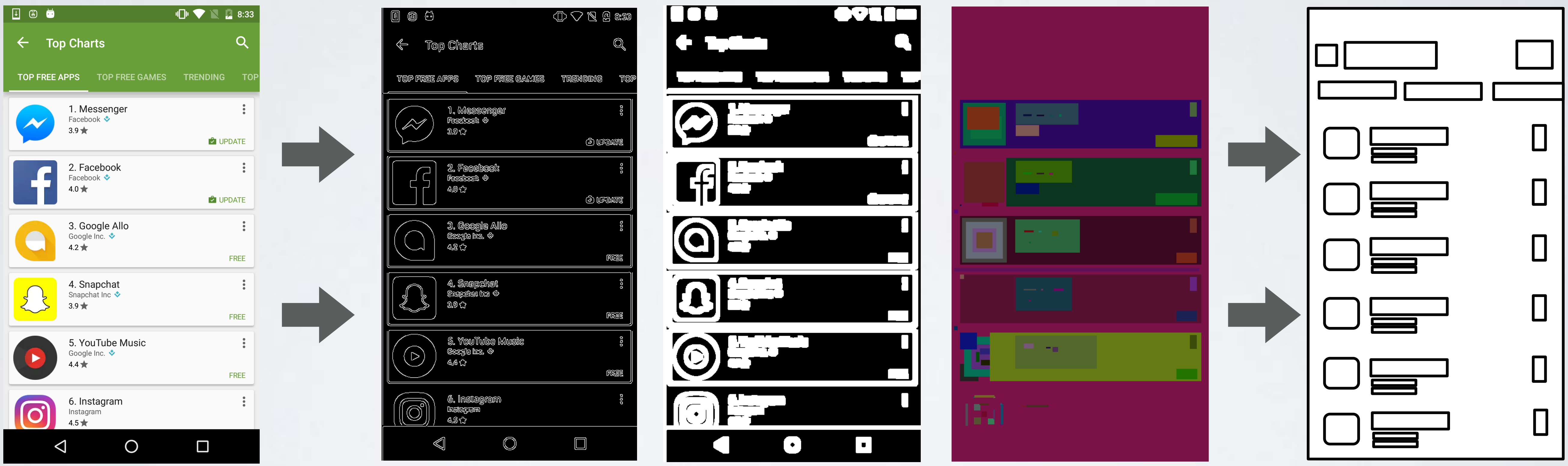
THE REDRAW FRAMEWORK



Phase 1: GUI-Component Detection

Computer Vision-Based Detection*

Credits to: Steven Walker & William Hollingsworth

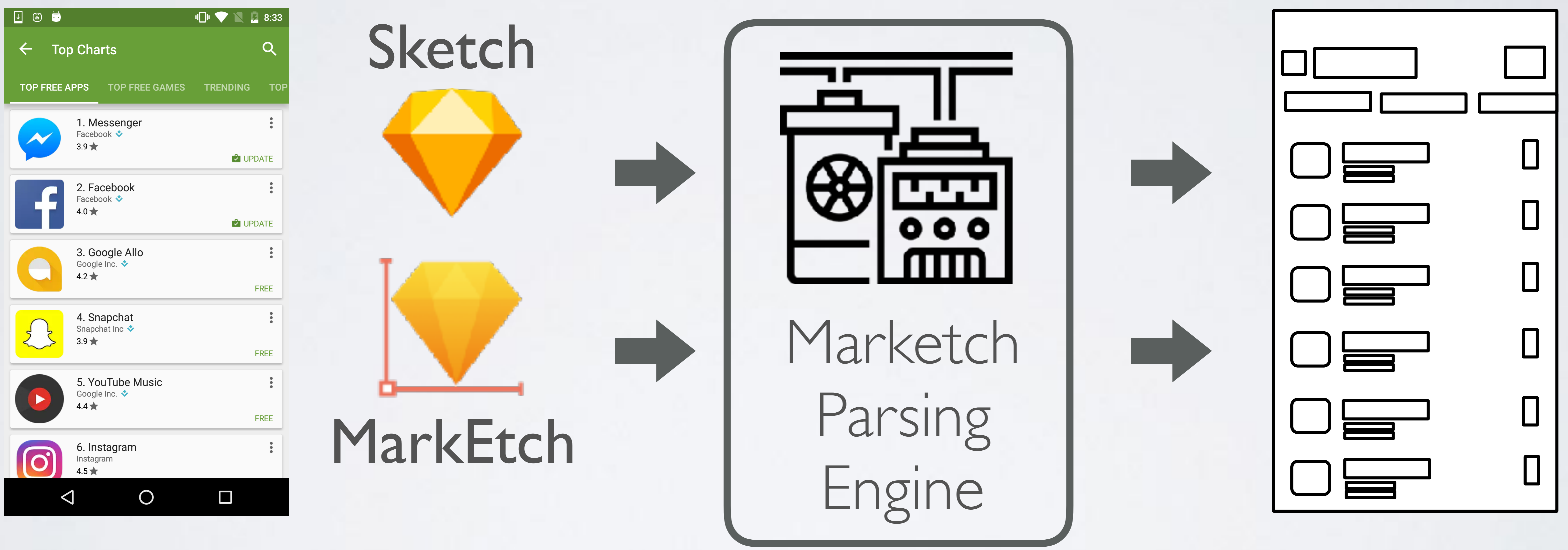


Input Image | Canny's Edge Detection | Edge Dilation | Contour Bounding Boxes | GUI-Component Bounding Boxes

*Tuan Anh Nguyen and Christoph Csallner. 2015. Reverse Engineering Mobile Application User Interfaces with REMAUI, In Proceedings of the 2015 30th IEEE/ACM International Conference on Automated Software Engineering (ASE '15)

Phase 1: GUI-Component Detection

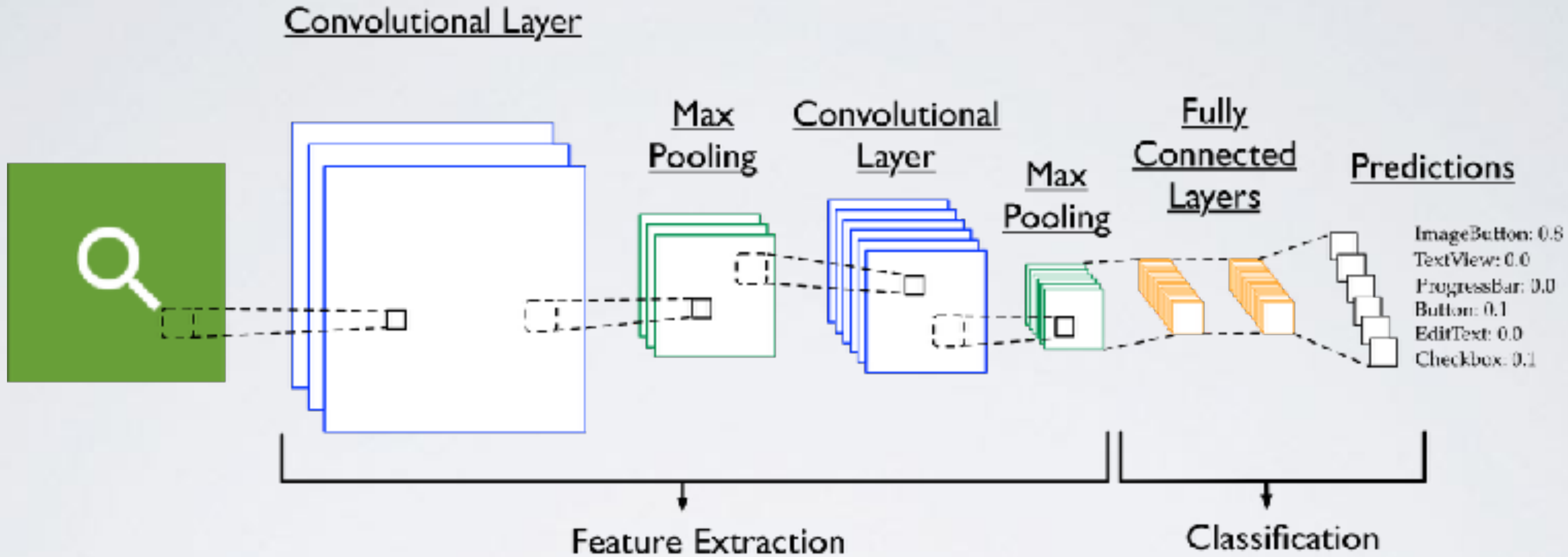
Parsing Mock-Up Metadata



Input Image

GUI-Component
Bounding Boxes

Phase 2: GUI-Component Classification

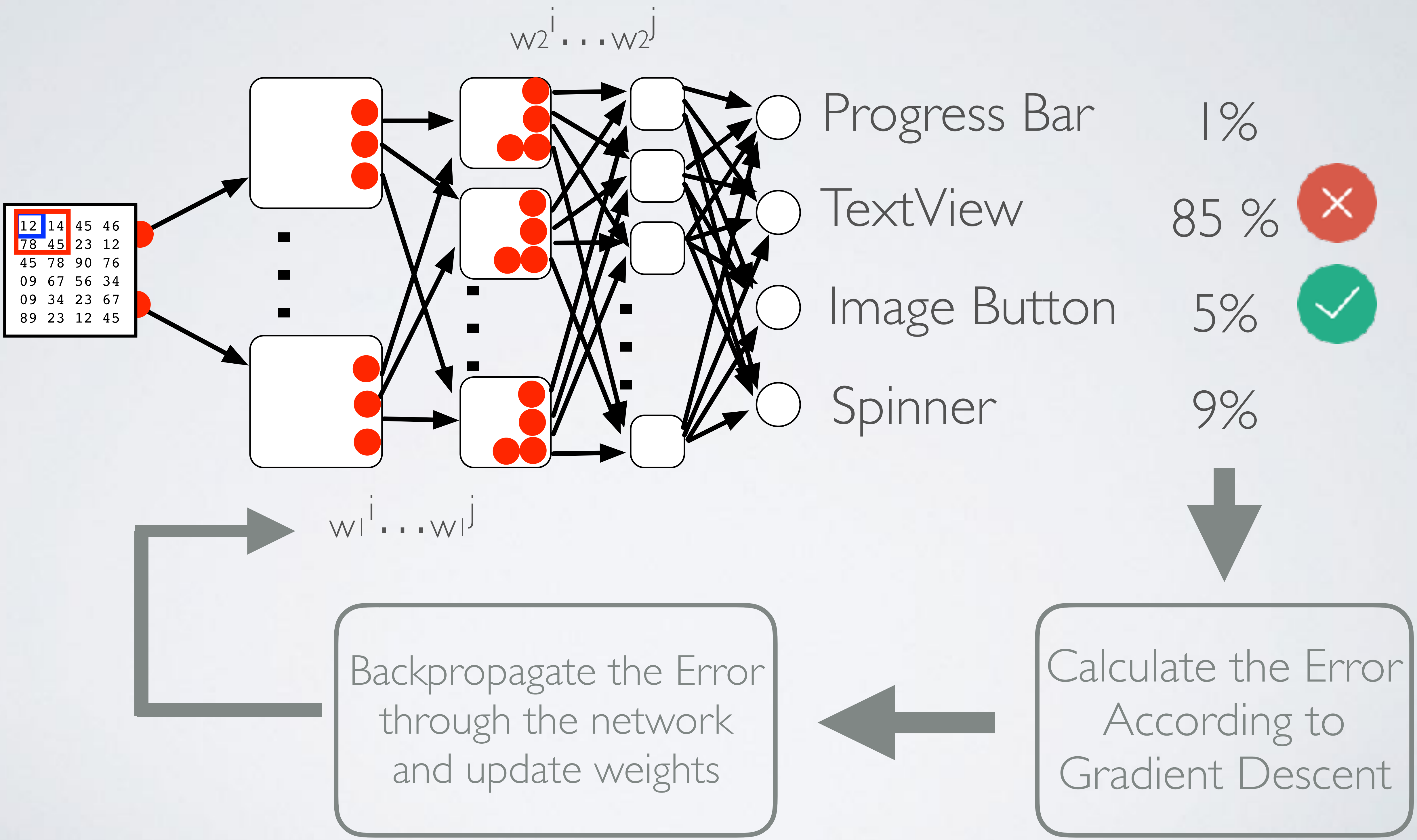


ConvNets or CNNs have revolutionized the task of image classification

Advanced approaches have reached human-levels of classification accuracy

Source: http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp/

Aside: CNN Basics



Phase 2: GUI-Component Classification

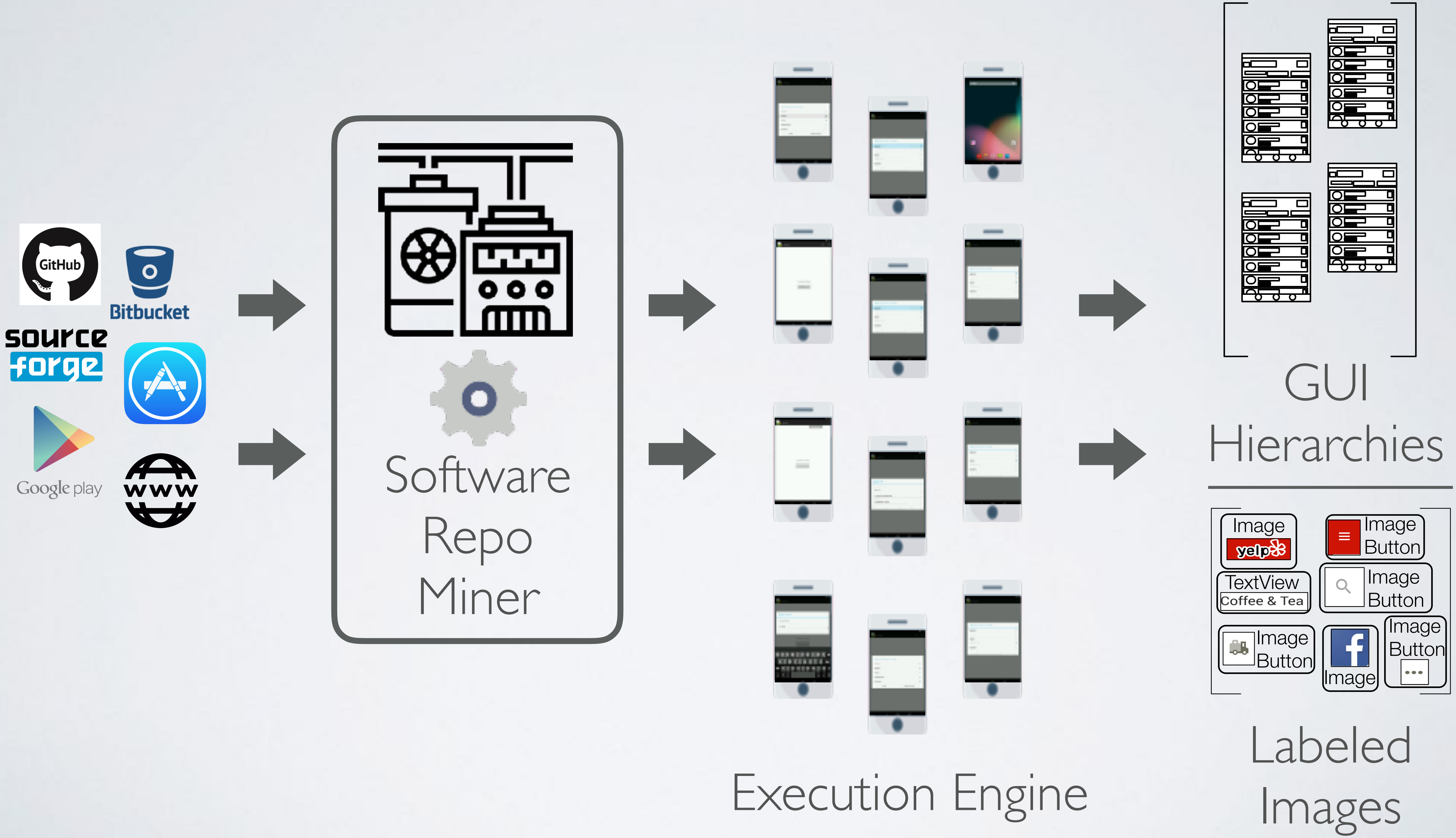
How can we extract a large enough dataset of training images?

Mining Software Repositories

+

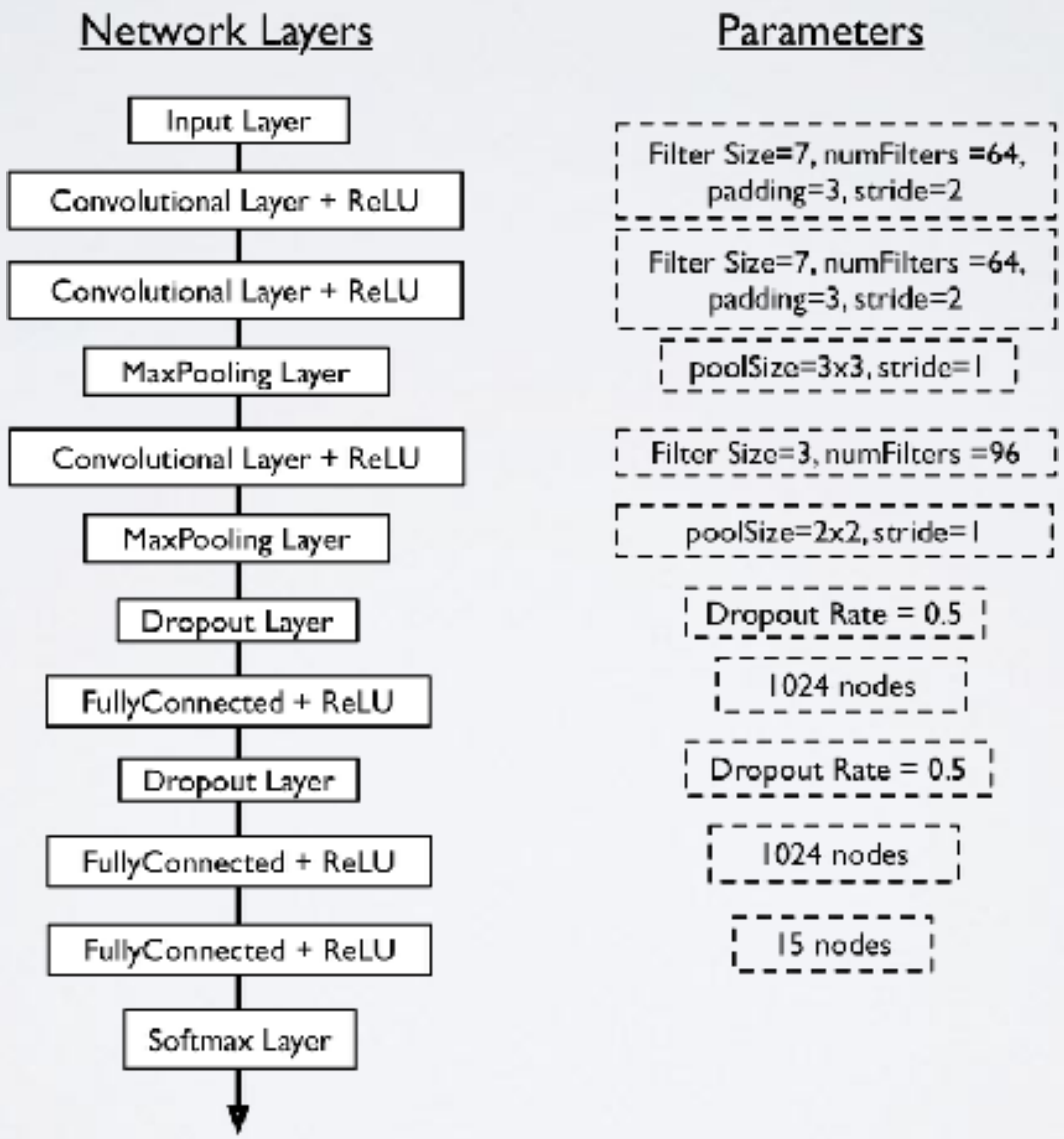
Automated Dynamic Analysis

PHASE 2: GUI-COMPONENT CLASSIFICATION

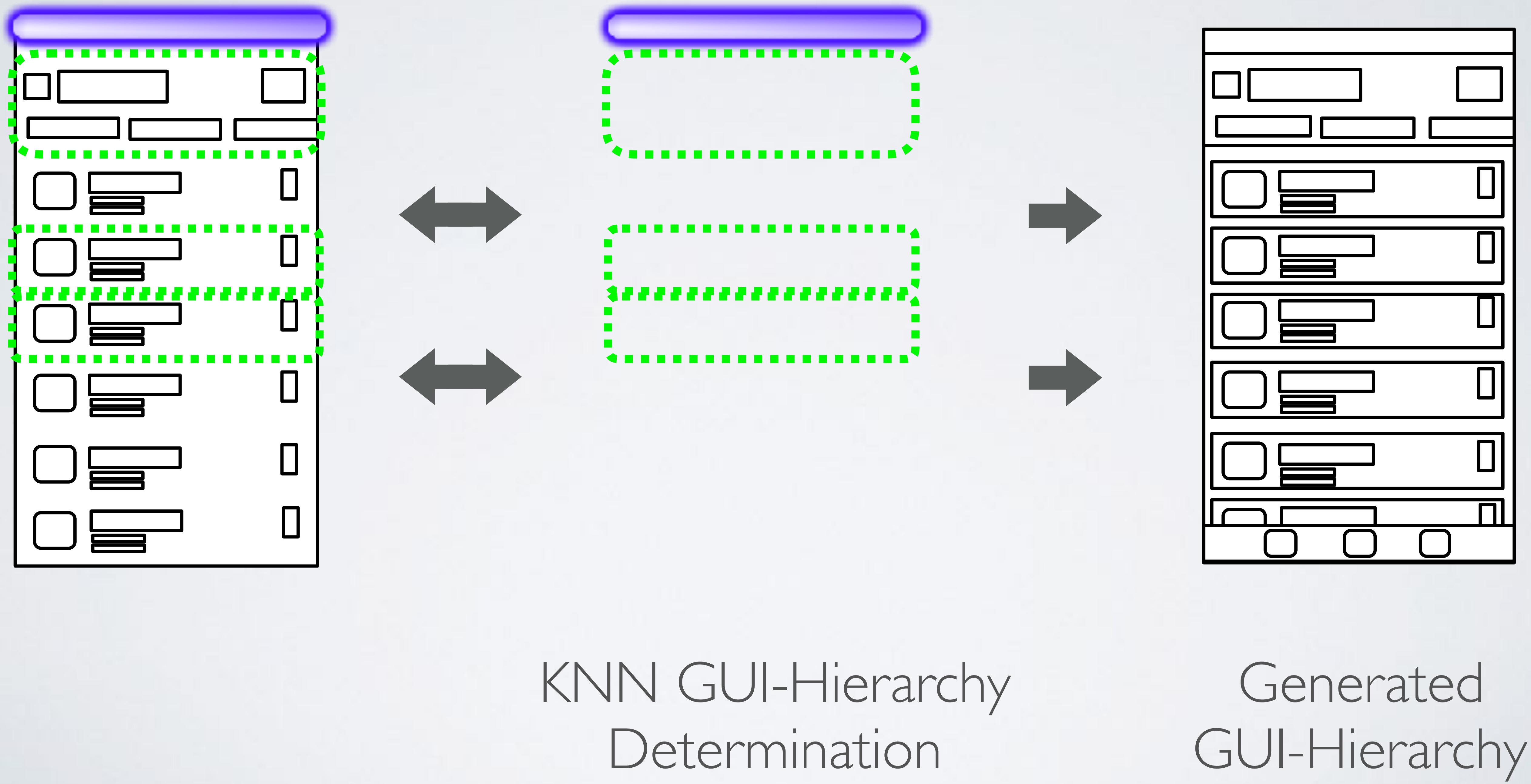


PHASE 2: GUI-COMPONENT CLASSIFICATION

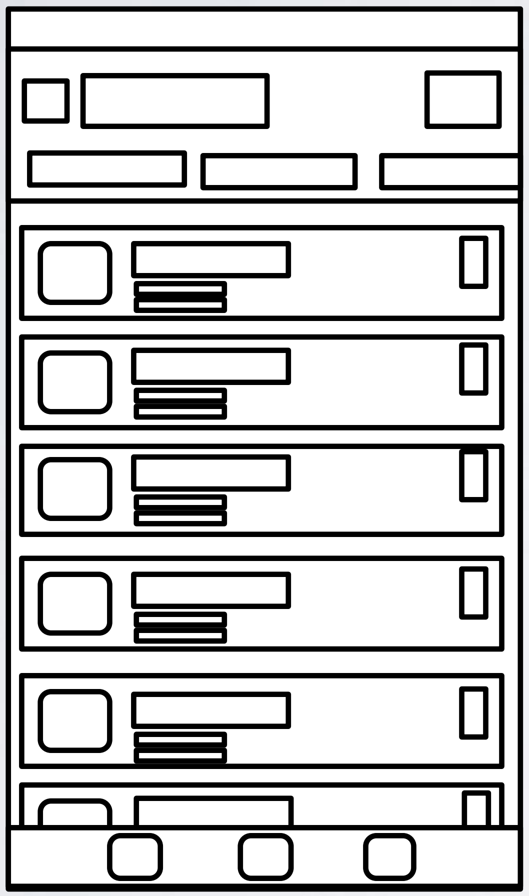
ReDraw's CNN Architecture



Phase 3: GUI-Hierarchy Construction



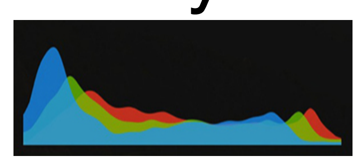
PHASE 3: APPLICATION ASSEMBLY

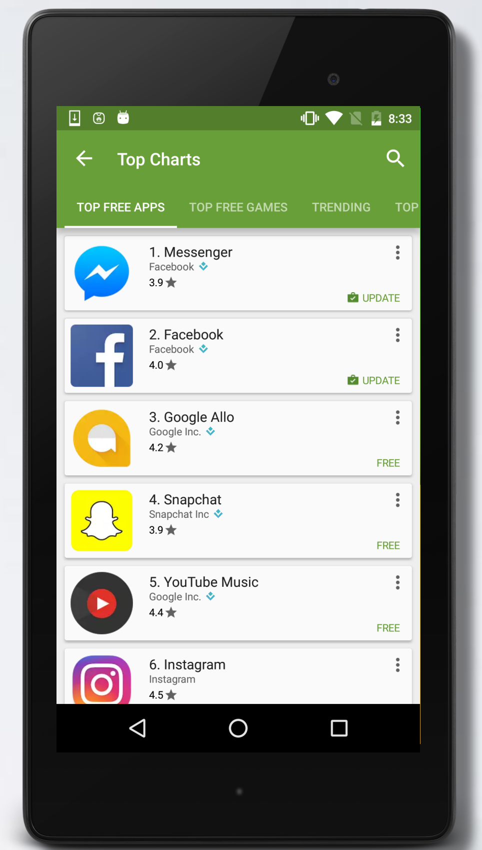


Labeled GUI Component Hierarchy

GUI-Code Generation

Color Quantization & Histogram Analysis





Fully Functional Prototype Application

Empirical Study

- **RQ1:** CNN Accuracy?
- **RQ2:** Hierarchy Similarity?
- **RQ3:** Visual Similarity?
- **RQ4:** Industrial Applicability?

Study 1: CNN Classifier Accuracy

Context

Google play

8,655 Applications

14,382 Unique Screens

191,300 Labeled GUI Component Images

ReDraw CNN Classifier

Supervised BoVW Baseline

Study 1:Results

	Total	TV	IV	Bt	S	ET	IBt	CTV	PB	RB	TBt	CB	Sp	SB	NP	RBt
TV	9877	59%	4%	9%	1%	6%	2%	8%	6%	0%	1%	2%	0%	1%	0%	2%
IV	5345	4%	51%	4%	1%	2%	11%	2%	18%	1%	1%	3%	0%	2%	0%	2%
Bt	1600	6%	6%	59%	1%	5%	4%	7%	4%	0%	1%	1%	0%	0%	3%	1%
S	37	5%	0%	3%	65%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	0%	0%	0%	0%	0%
ET	567	6%	2%	4%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	1%	0%	0%	4%	1%
IBt	866	2%	16%	3%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1%	2%	0%	1%	0%	3%
CTV	337	3%	1%	7%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	2%	0%	0%	0%	2%
PB	41	0%	24%	2%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	2%	2%	2%	0%	2%
RB	22	0%	5%	0%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	0%	0%	0%	0%	0%
TBt	26	7%	7%	19%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	33%	0%	0%	0%	0%	7%
CB	165	4%	2%	3%	1%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	72%	0%	0%	0%	1%
Sp	2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	10%0	0%	0%	0%
SB	39	0%	5%	0%	0%	0%	0%	0%	18%	3%	0%	5%	0%	68%	0%	3%
NP	40	3%	0%	5%	0%	3%	0%	5%	0%	0%	0%	0%	0%	0%	84%	0%
RBt	129	6%	3%	5%	1%	3%	0%	6%	18%	0%	1%	1%	0%	1%	0%	55%

Overall Accuracy

65%

Confusion Matrix for BoVW Baseline

Study 1: Results

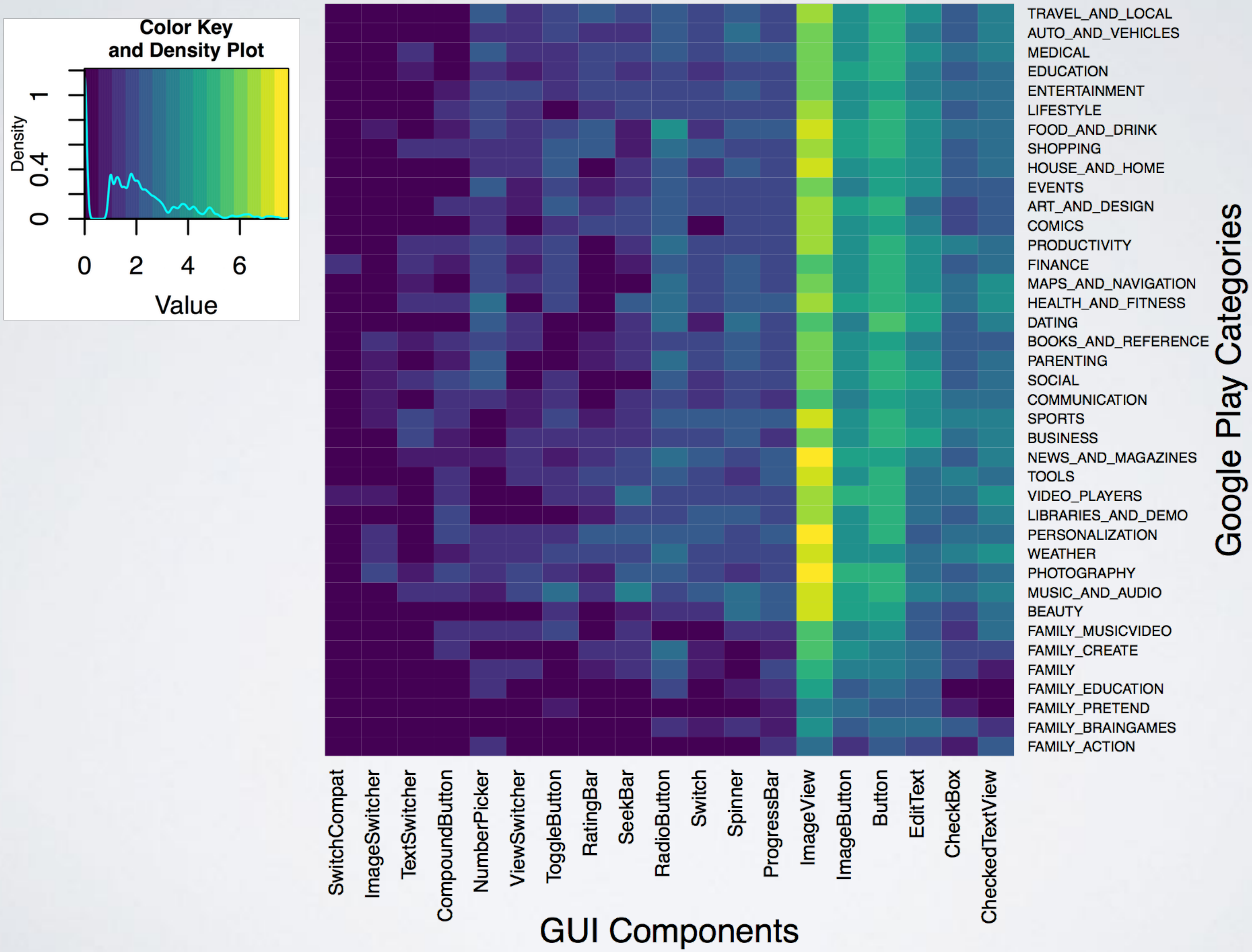
	Total	TV	IV	Bt	S	ET	IBt	CTV	PB	RB	TBt	CB	Sp	SB	NP	RBt
TV	9877	94%	3%	2%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%
IV	5345	5%	93%	1%	0%	0%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%
Bt	1600	11%	6%	81%	0%	1%	1%	0%	0%	0%	0%	0%	0%	0%	0%	0%
S	37	5%	3%	0%	87%	0%	0%	5%	0%	0%	0%	0%	0%	0%	0%	0%
ET	567	14%	3%	2%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	0%	0%	0%	0%	0%
IBt	866	4%	23%	1%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	0%	0%	0%	0%	0%
CTV	337	7%	0%	0%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	0%	0%	0%	0%	0%
PB	41	15%	29%	0%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	0%	0%	0%	0%	0%
RB	22	0%	0%	0%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0%	0%	0%	0%	0%	0%
TBt	26	19%	22%	7%	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	52%	0%	0%	0%	0%	0%
CB	165	12%	7%	0%	0%	1%	0%	0%	0%	0%	0%	81%	0%	0%	0%	0%
Sp	2	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	100%	0%	0%	0%
SB	39	10%	13%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	78%	0%	0%
NP	40	0%	5%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	0%	95%	0%
RBt	129	4%	3%	2%	0%	0%	0%	1%	0%	0%	0%	1%	0%	0%	0%	89%

Overall Accuracy

91%

Confusion Matrix for ReDraw CNN

STUDY 1: CNN CLASSIFIER ACCURACY



STUDY 2:GUI HIERARCHY CONSTRUCTION

Context

Google play

32 Applications

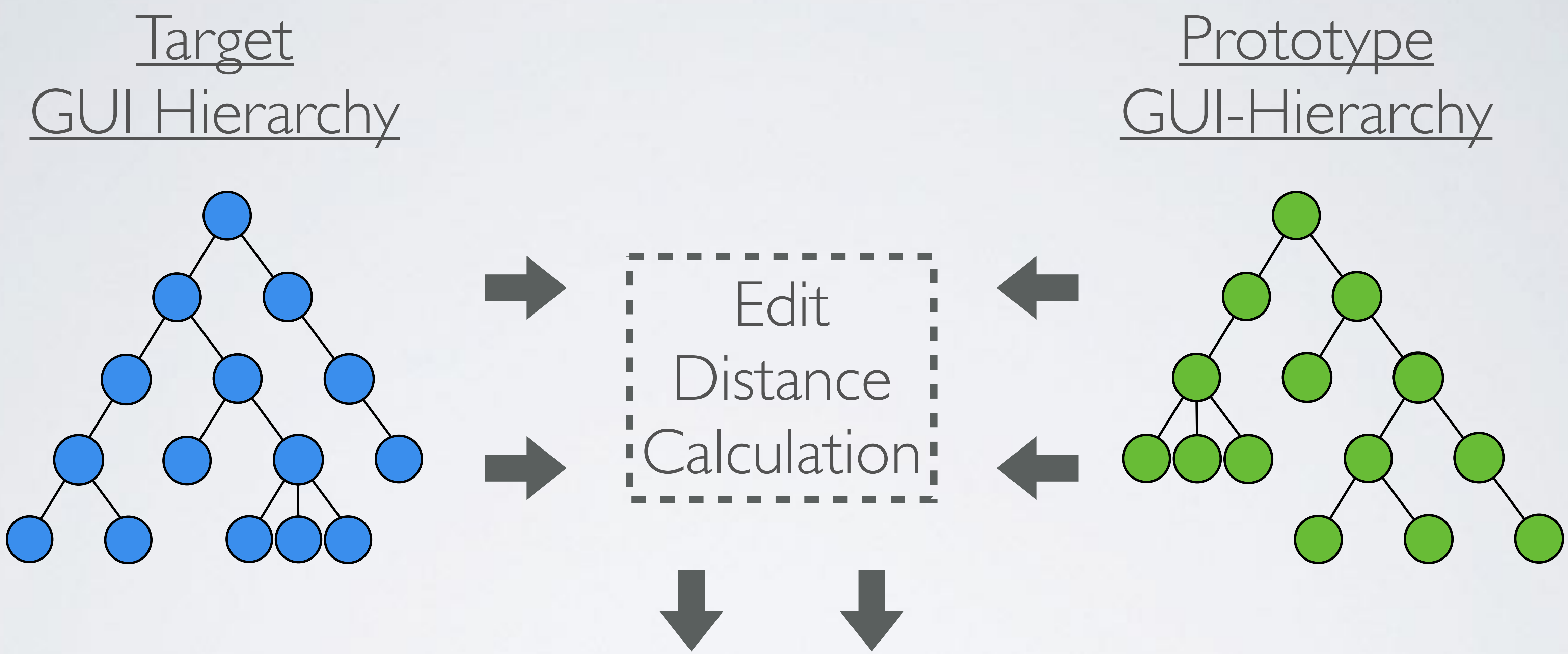
2-3 Unique Screens (Not used in Training the CNN)

ReDraw

Pix2Code

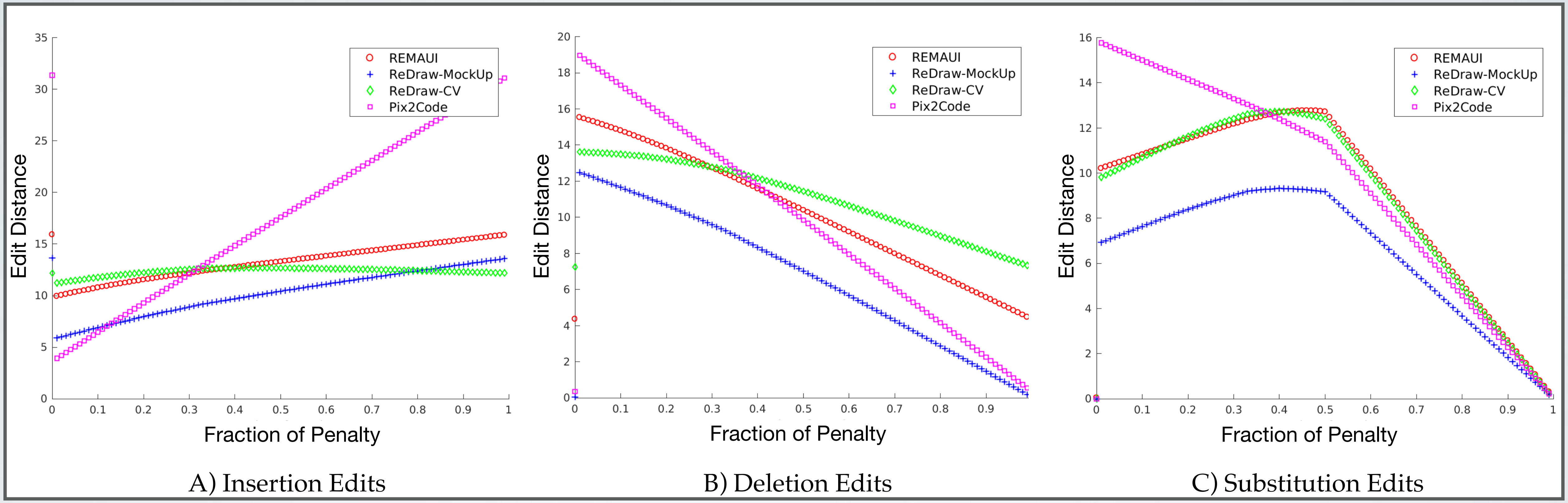
REMAUI (Re-implementation)

Study 2:GUI Hierarchy Construction



# of Insertions - 0 < Insertion Penalty < 1
# of Deletions - 0 < Deletion Penalty < 1
# of Substitutions - 0 < Substitution Penalty < 1

STUDY 2: RESULTS



Hierarchy Edit Distance to Ground Truth

Study 3: Visual Similarity of Apps

Context

Google play

32 Applications

2-3 Unique Screens (Not used in Training the CNN)

ReDraw

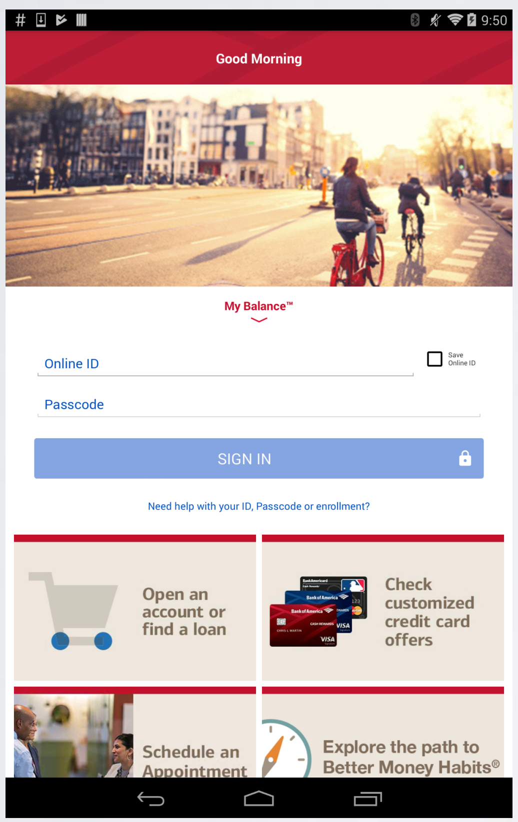
Pix2Code

REMAUI (Re-implementation)

Study 3: Visual Similarity of Apps

Target App Screenshot

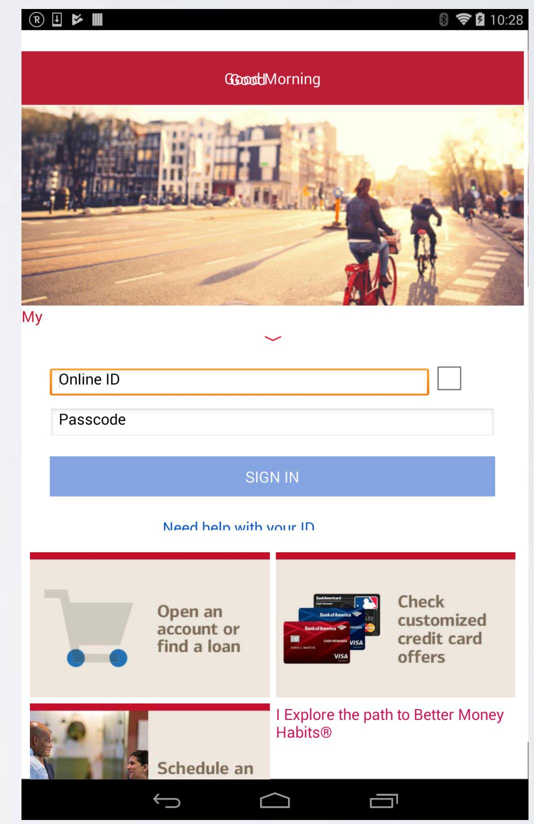
Generated App Screenshot



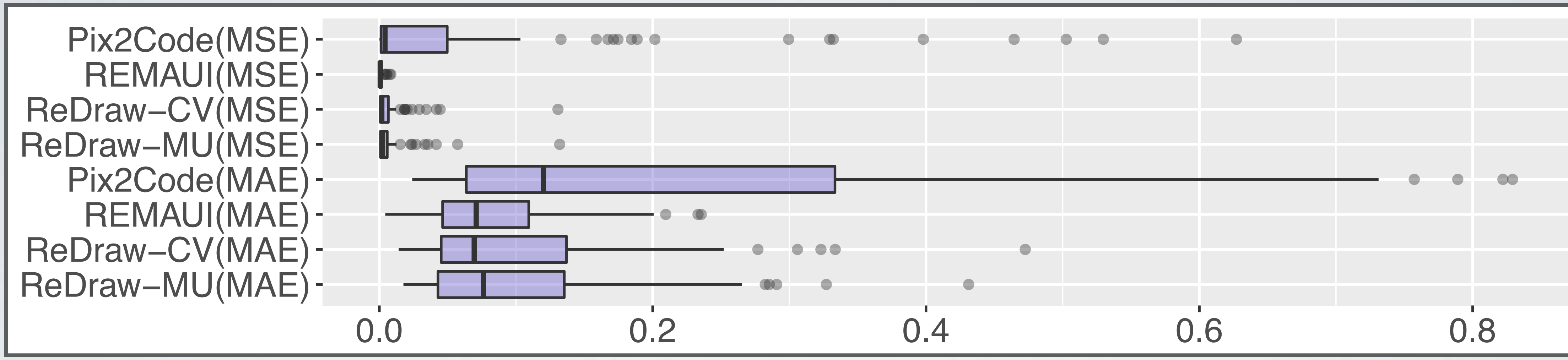
Pixel-Based

Mean Average Error (MAE)

Mean Squared Error (MSE)



Study 3: Results



MSE and MAE compared to Ground Truth Screenshots

MAE

Test	*p*-Value	*d*
ReDrawMU vs. ReDrawCV	0.835	0.02 (Small)
ReDrawMU vs. REMAUI	0.542	0.06 (Small)
ReDrawMU vs. pix2code	<0.0002	-0.34 (Medium)
pix2code vs. ReDrawCV	<0.0001	0.35 (Medium)
pix2code vs. REMAUI	<0.0001	0.39 (Medium)
REMAUI vs. ReDrawCV	0.687	-0.04 (Small)

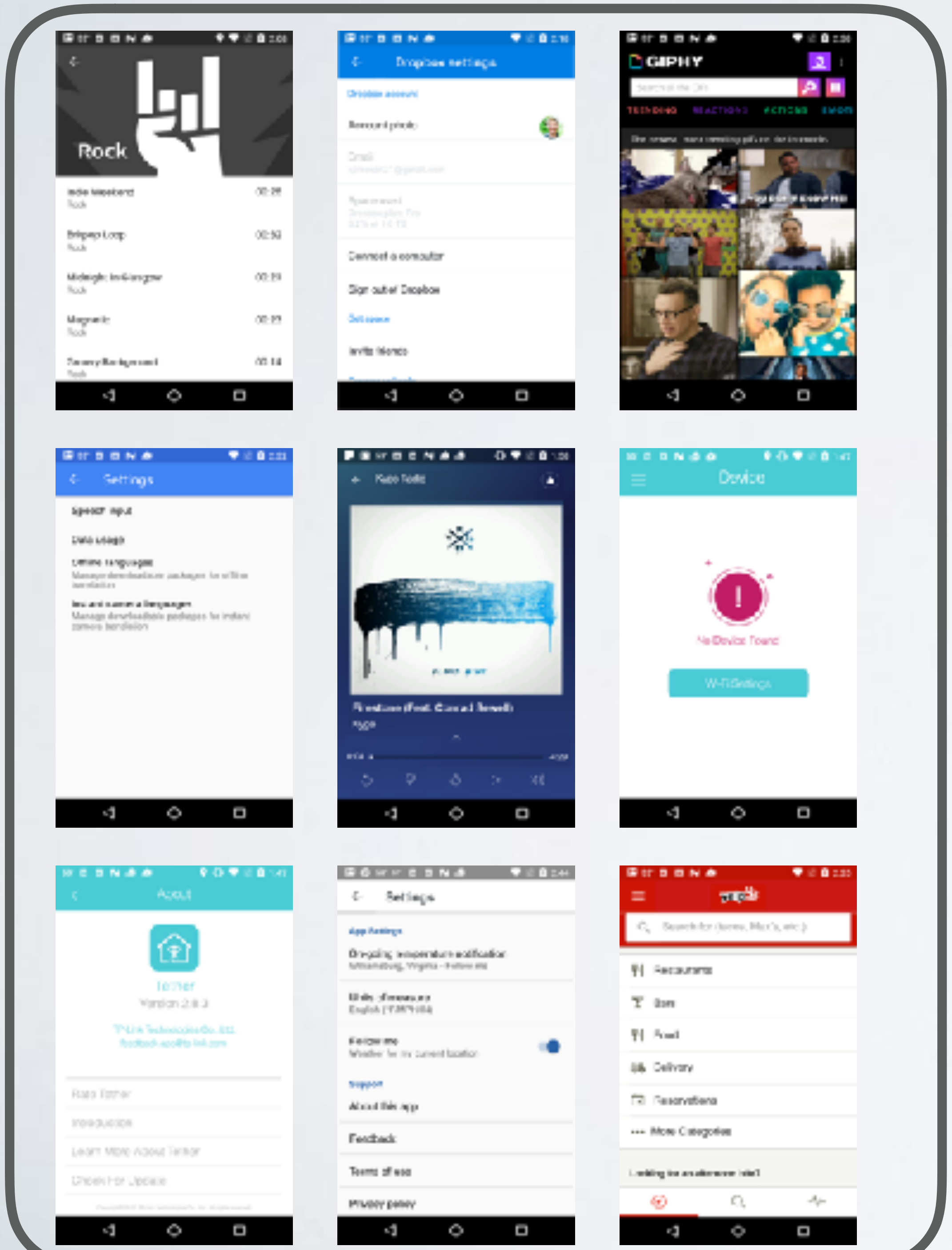
MSE

Test	*p*-Value	*d*
ReDrawMU vs. ReDrawCV	0.771	0.03 (Small)
ReDrawMU vs. REMAUI	<0.0001	0.45 (Medium)
ReDrawMU vs. pix2code	<0.003	-0.27 (Small)
pix2code vs. ReDrawCV	<0.002	0.28 (Small)
pix2code vs. REMAUI	<0.0001	0.61 (Large)
REMAUI vs. ReDrawCV	<0.0001	-0.42 (Medium)

Mann Whitney U-Test & Cliff's Delta

Study4: Developer Utility

9 Screens from 8 Popular Apps



Context

Mobile Front-End Developer

Google

Mobile Designer

HUAWEI

Mobile Researcher

facebook

STUDY4: RESULTS

“It’s a good starting point... From a development standpoint, the thing I would appreciate most is getting a lot of the boilerplate code done [automatically]”

“There are going to be edge cases for different layouts, but these are easily fixed after the fact”

“The key thing is fast iteration. A developer could generate the initial view [using ReDraw], clean up the layouts, and have a working app. If a designer could upload a screenshot, and without any other intervention [ReDraw] could update the [existing] xml this would be ideal.”

APPLICATIONS GENERATED BY REDRAW

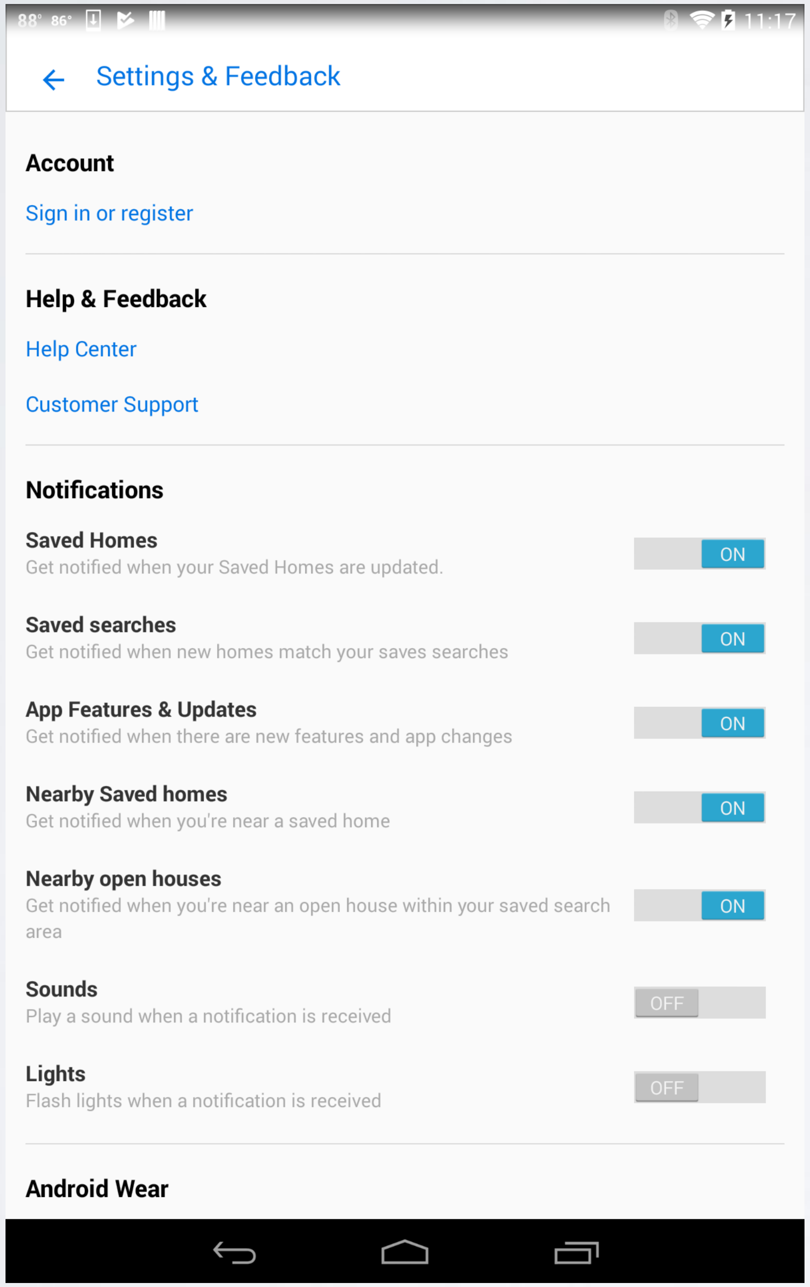
Textgram

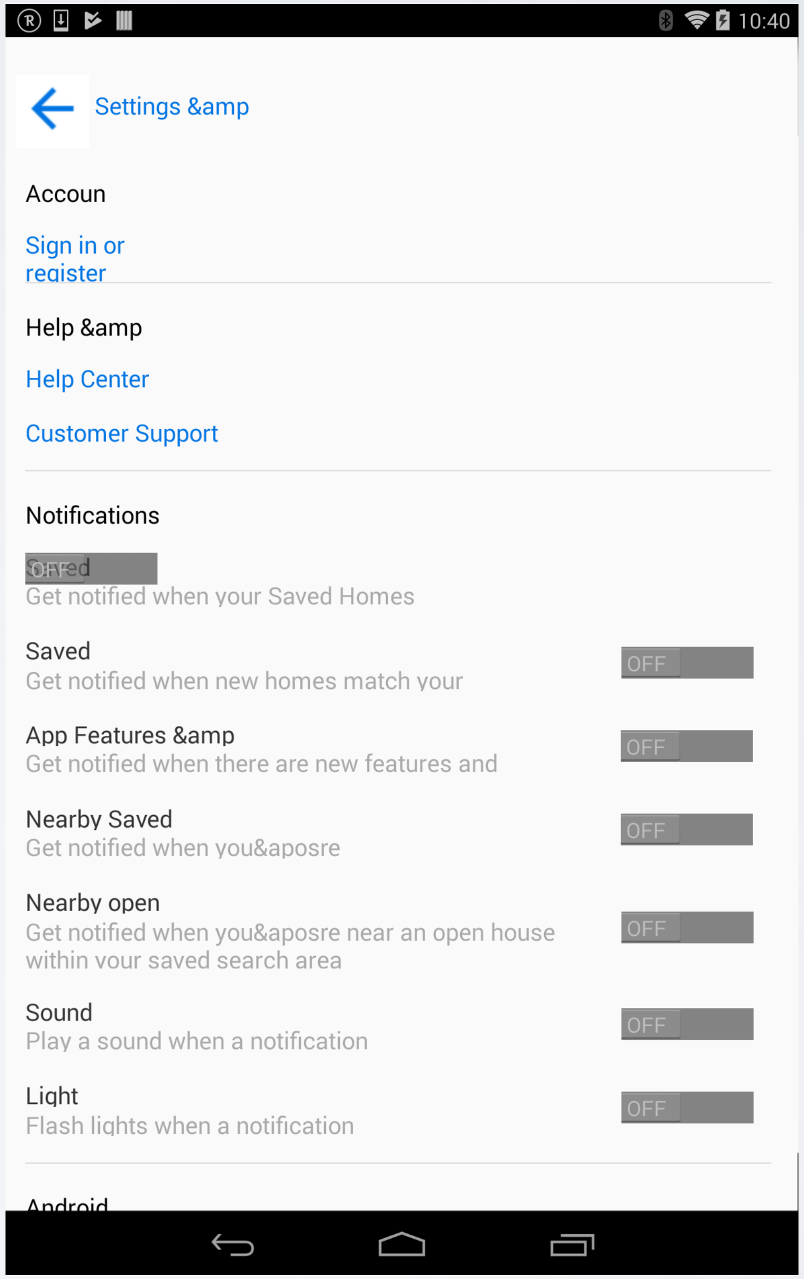
A) Original Application

B) ReDraw App (MockUp)

Applications Generated By ReDraw

Zillow



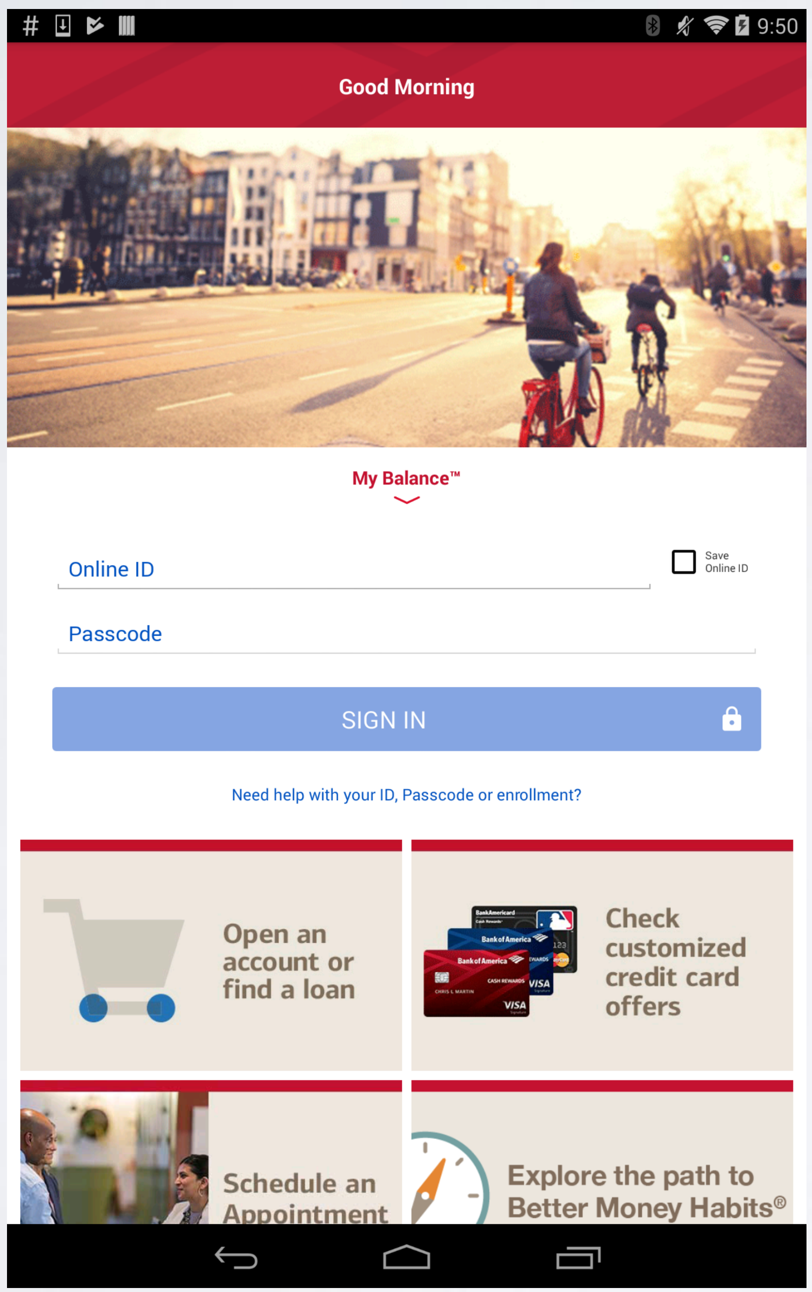


A) Original Application

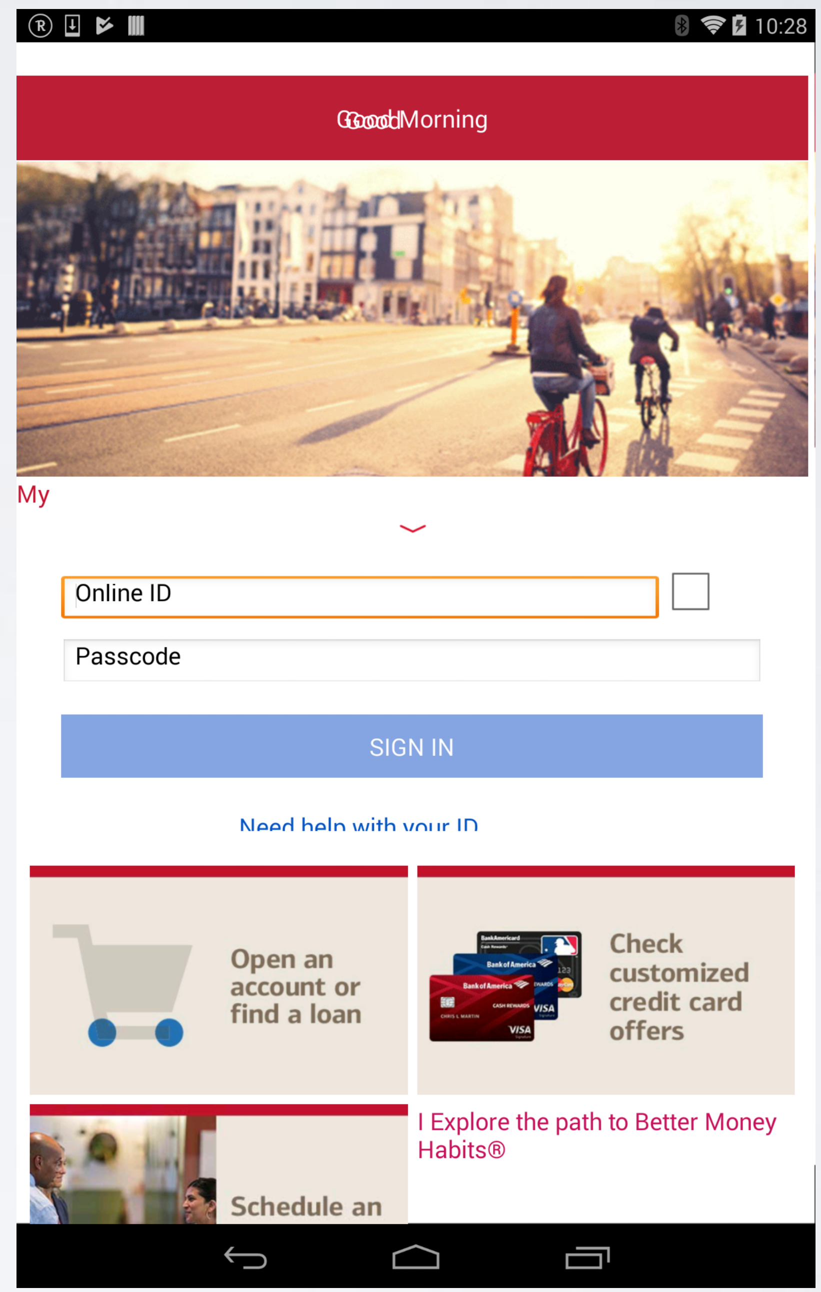
B) ReDraw App (MockUp)

Applications Generated By ReDraw

Bank of America



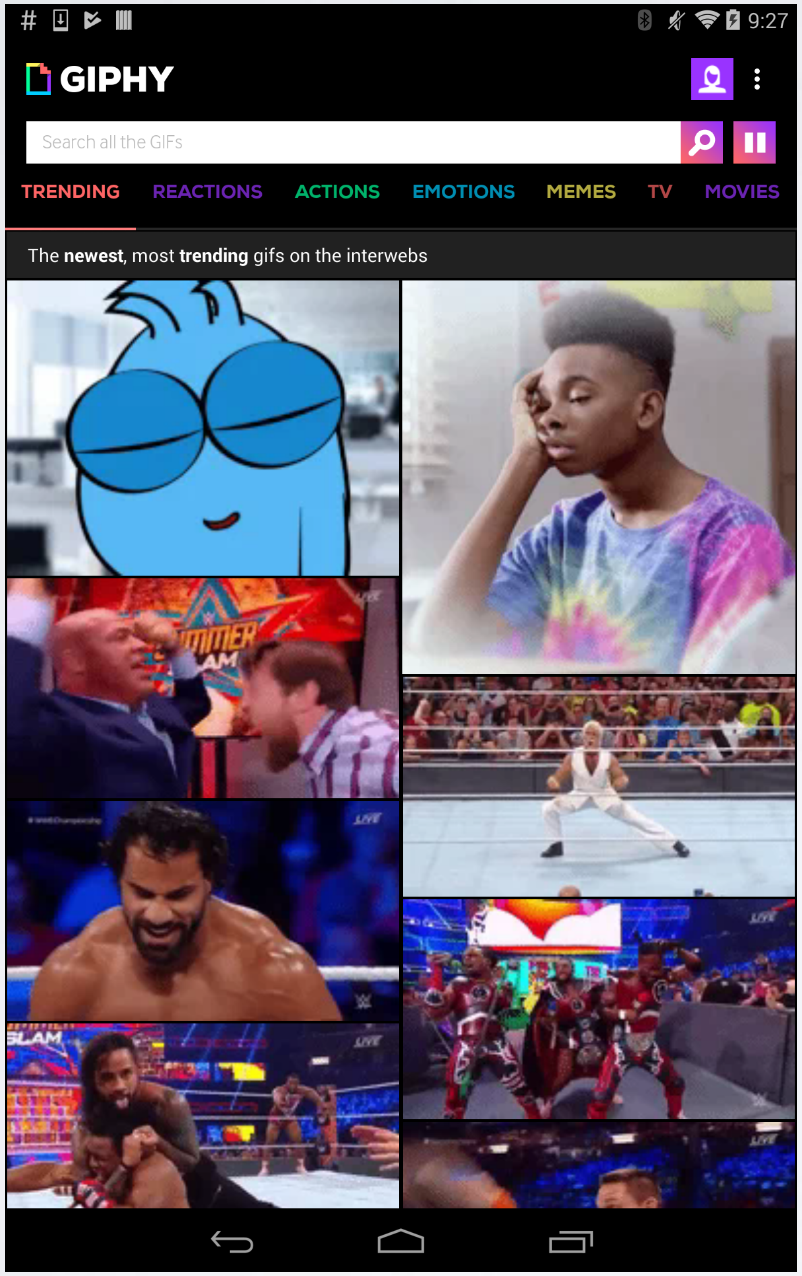
A) Original Application



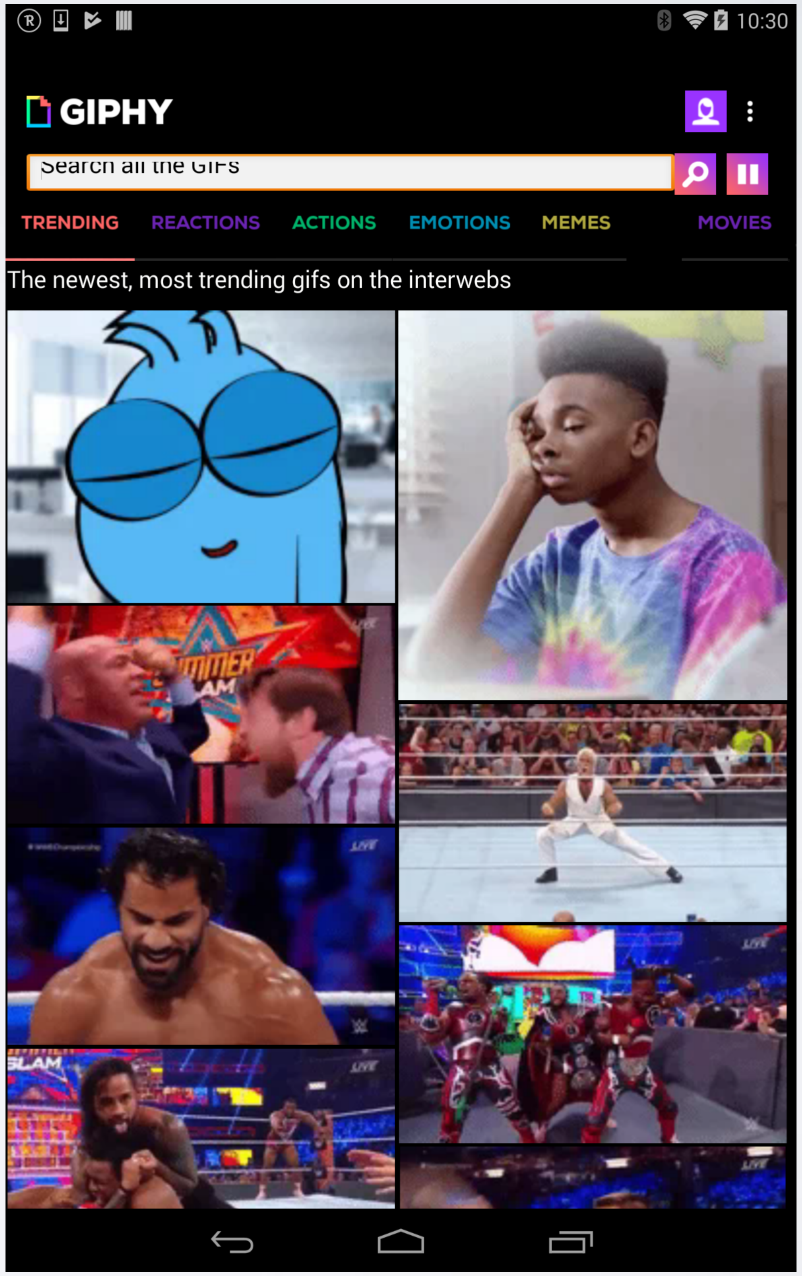
B) ReDraw App (MockUp)

Applications Generated By ReDraw

Giphy



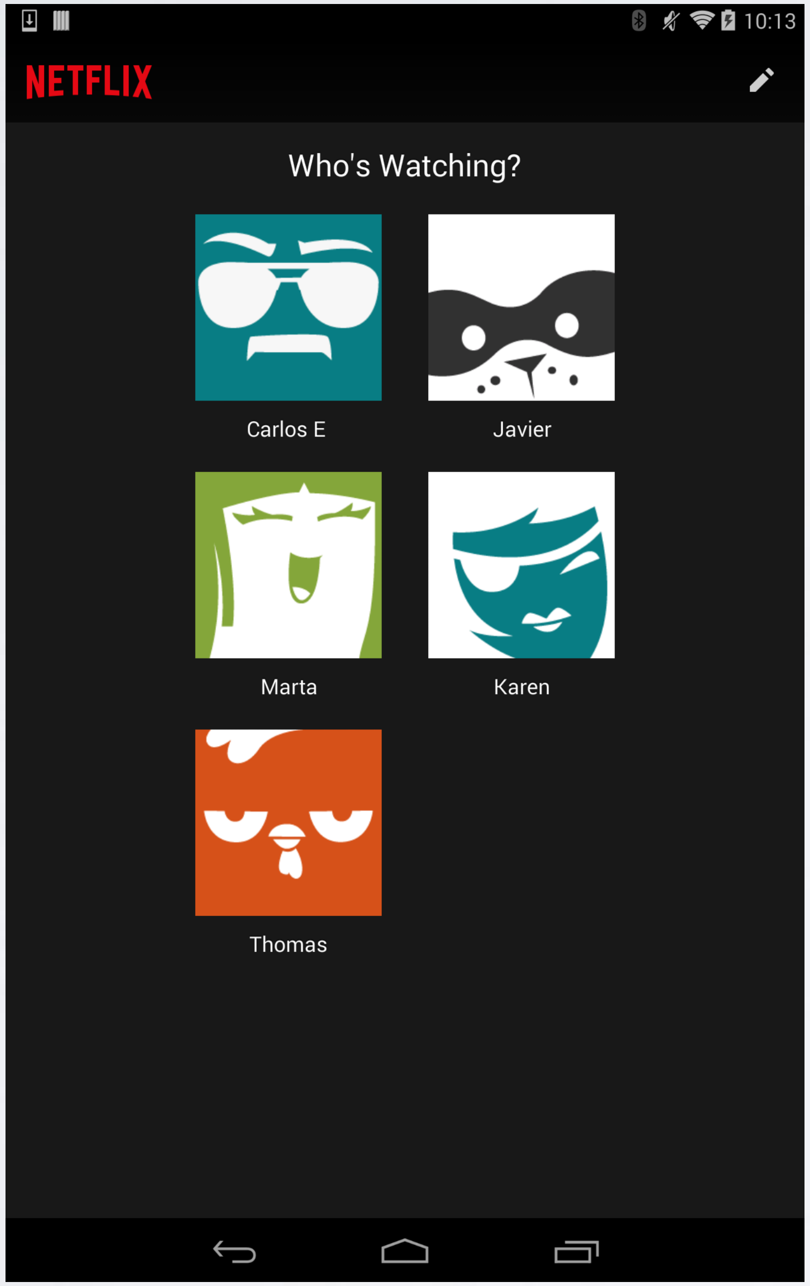
A) Original Application



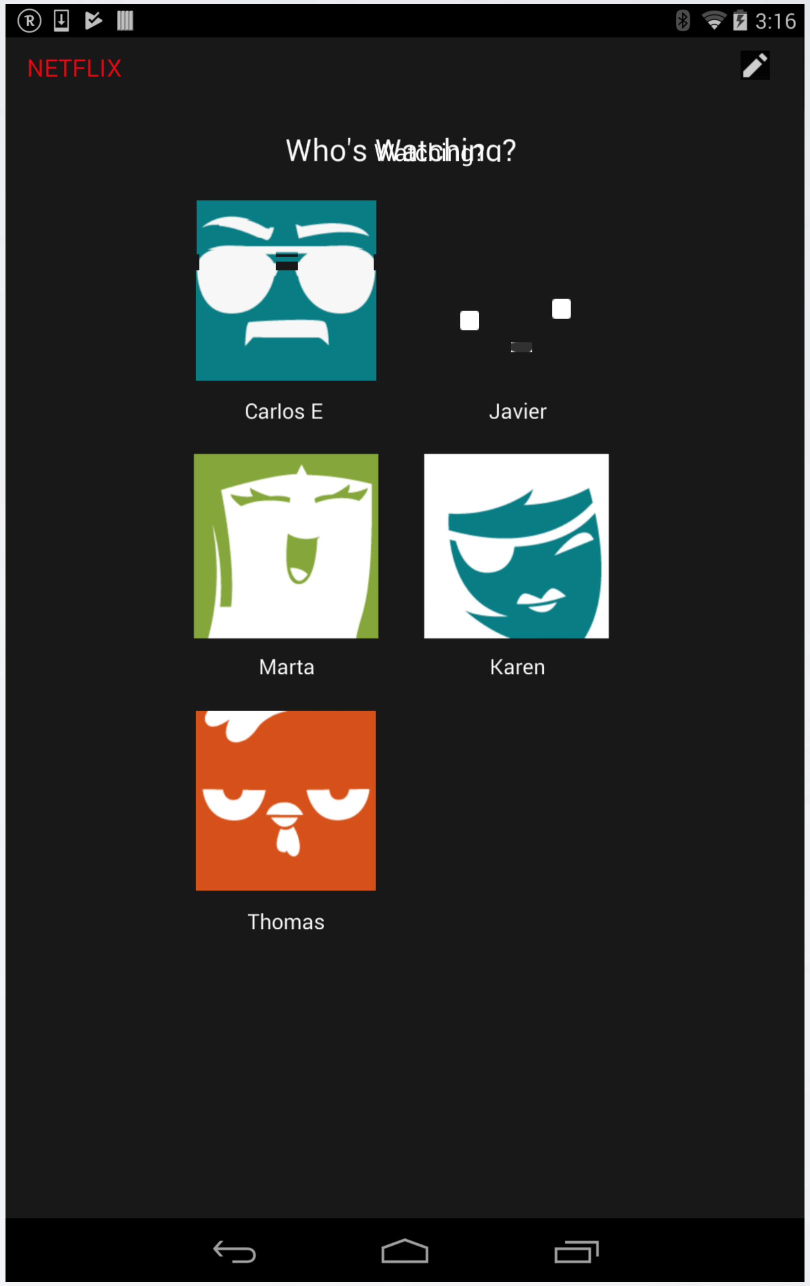
B) ReDraw App (CV)

Applications Generated By ReDraw

Netflix



A) Original Application



B) ReDraw App (CV)

ACCEPTED AT IEEE TSE'18

Transactions on Software Engineering

The ***IEEE Transactions on Software Engineering* (TSE)** is an archival journal published bimonthly. We are interested in well-defined theoretical results and empirical studies that have potential impact on the construction, analysis, or management of software. Read the full scope of TSE.

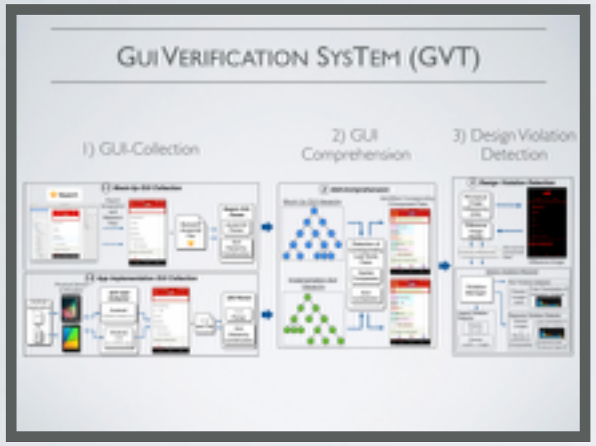
Machine Learning-Based Prototyping of Graphical User Interfaces for Mobile Apps

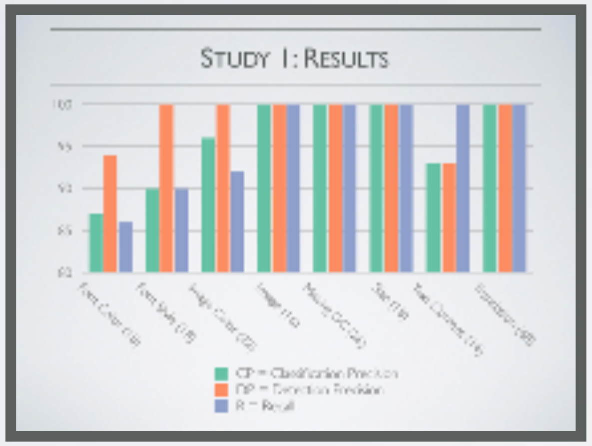
Kevin Moran, *Student Member, IEEE*, Carlos Bernal-Cárdenas, *Student Member, IEEE*, Michael Curcio, *Student Member, IEEE*, Richard Bonett, *Student Member, IEEE*, and Denys Poshyvanyk, *Member, IEEE*

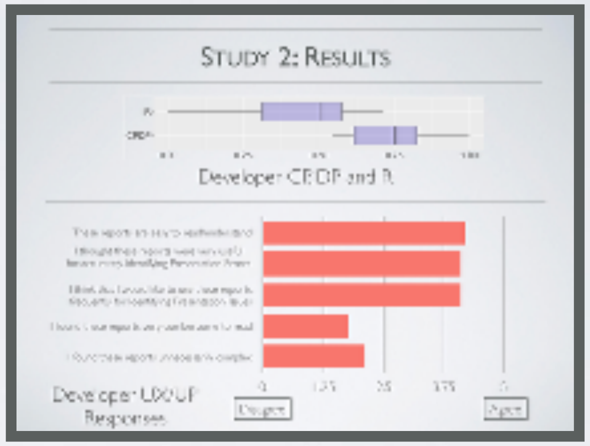
Abstract—It is common practice for developers of user-facing software to transform a mock-up of a graphical user interface (GUI) into code. This process takes place both at an application's inception and in an evolutionary context as GUI changes keep pace with evolving features. Unfortunately, this practice is challenging and time-consuming. In this paper, we present an approach that automates this process by enabling accurate prototyping of GUIs via three tasks: *detection*, *classification*, and *assembly*. First, logical components of a GUI are *detected* from a mock-up artifact using either computer vision techniques or mock-up metadata. Then, software repository mining, automated dynamic analysis, and deep convolutional neural networks are utilized to accurately *classify* GUI-components into domain-specific types (*e.g.*, toggle-button). Finally, a data-driven, K-nearest-neighbors algorithm generates a suitable hierarchical GUI structure from which a prototype application can be automatically *assembled*. We implemented this approach for Android in a system called REDRAW. Our evaluation illustrates that REDRAW achieves an average GUI-component classification accuracy of 91% and assembles prototype applications that closely mirror target mock-ups in terms of visual affinity while exhibiting reasonable code structure. Interviews with industrial practitioners illustrate ReDraw's potential to improve real development workflows.

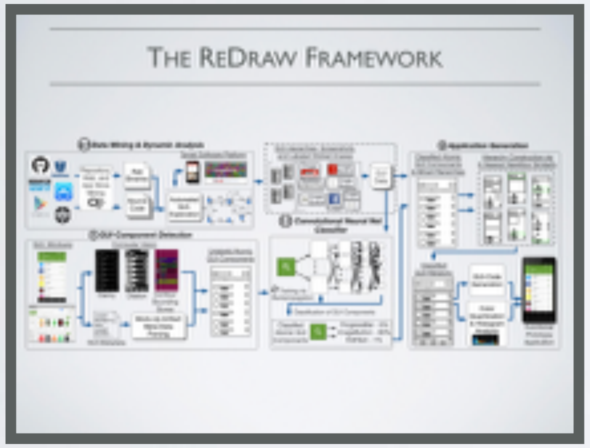
Index Terms—GUI, CNN, Mobile, Prototyping, Machine-Learning, Mining Software Repositories.

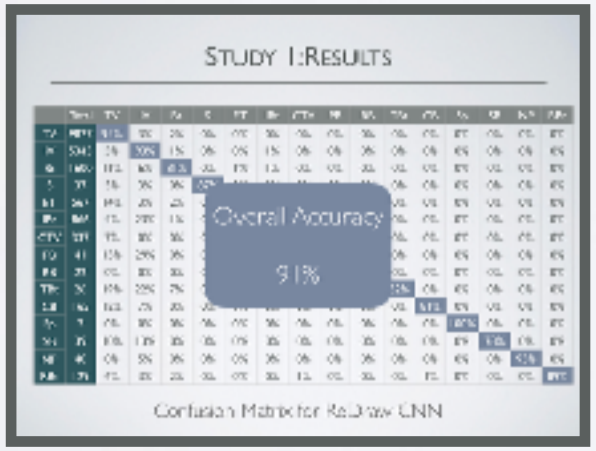
Conclusions

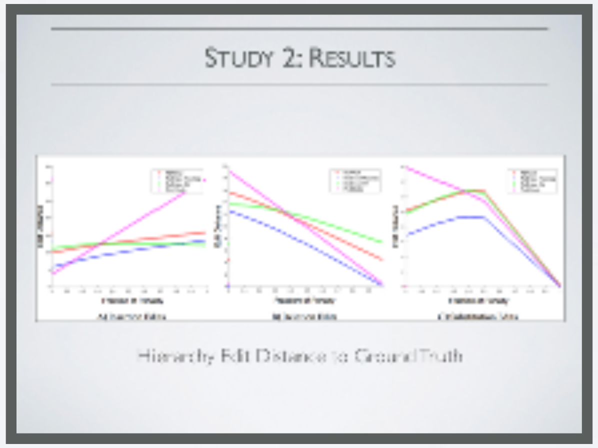


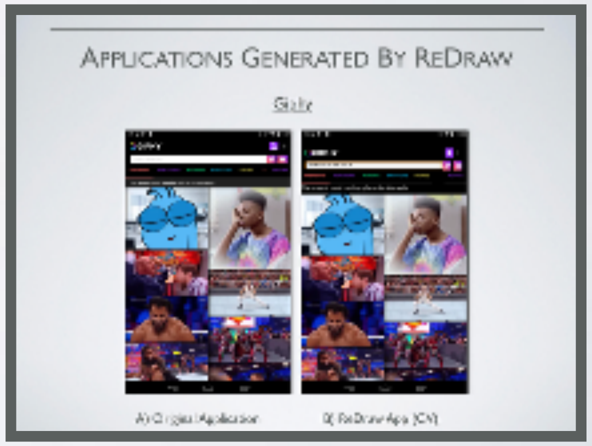










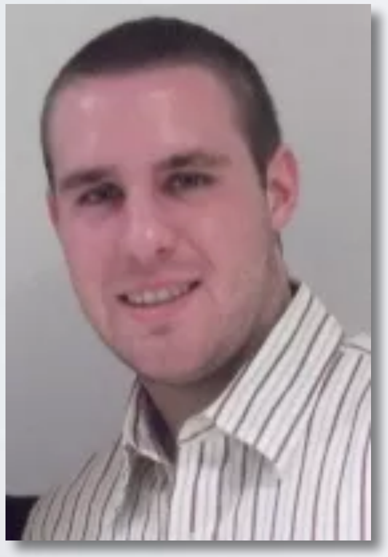


My Android Team

Mario

Kevin

Richie



Chris

Carlos

Michele

ACKNOWLEDGEMENTS

- We gratefully acknowledge NSF for supporting this research
- W&M Plumeri Award

Thank you!

Any Questions?

Denys Poshyvanyk, Ph.D.
Associate Professor
denys@cs.wm.edu
http://www.cs.wm.edu/~denys