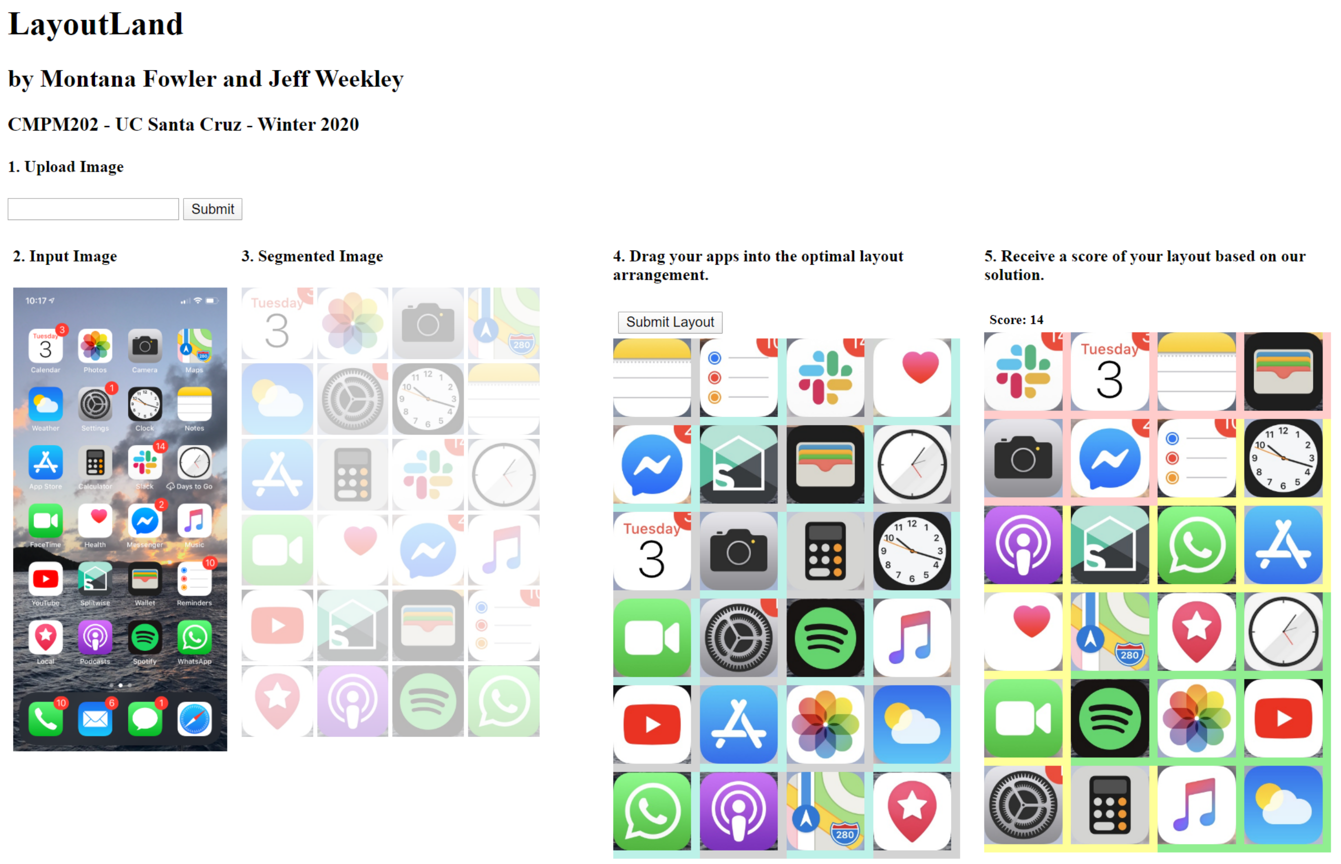
Layout Land: An Intelligent, Computer-aided Layout Decision Support System for iPhones

Montana Fowler
Jeffrey Weekley
montana_fowler@ucsc.edu
jweekley@ucsc.edu
University of California Santa Cruz
Santa Cruz, California



LayoutLand

by Montana Fowler and Jeff Weekley

CMPM202 - UC Santa Cruz - Winter 2020

1. Upload Image

2. Input Image

3. Segmented Image

4. Drag your apps into the optimal layout arrangement.

5. Receive a score of your layout based on our solution.

Score: 14

Figure 1: Layout Land User Interface for an auto-generated layout of iPhone screens

ABSTRACT

There are billions of possible applications (apps) for mobile devices. Curating their layout on small screens is often an afterthought. As early as the mid-1990's designers were considering the aesthetic effects of moving from large screens to small screens [6]. The shift from large screen to small screen impacts not only design, but also user interaction. Many of the interaction models expected for good human computer interaction (HCI) are present in small screens, but HCI is not sufficient. Three key elements have emerged in addition to good HCI as important factors when considering screen

layout: balance, unity, and sequence [2]. In this paper, we present an AI-enabled, game-based and personalized tool called Layout Land that serves as a user decision aid to help make better layout choices for their mobile device. The Layout Land system takes into consideration user preferences, heuristics and game theory to provide examples and incentives to the user for better layout of their most frequently used applications while maintaining balance, unity and sequence overall.

https://github.com/montanafowler/LayoutLand

CCS CONCEPTS

• **XXXX**;

KEYWORDS

image segmentation, neural networks, human-computer interaction, serious games

ACM Reference Format:
Montana Fowler and Jeffrey Weekley. 2020. Layout Land: An Intelligent, Computer-aided Layout Decision Support System for iPhones. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Early designers considered the implications of small screen layouts versus the more free-form layout possibilities of desktop computers, and the general solution was to constrain the layout of mobile devices to a grid. In this paper, we examine the layout of the current iterations of iPhones, which are constrained to a 4 x 6 matrix of application icons on the main screen. We do not consider secondary screens, or the persistent row of 1 x 4 icons in the "tray" at the bottom of the screen. Nor do we consider the ability of users to stack icons in pop-up sub-screens, though the general principles of balance, unity and sequence still apply in these areas.

While the constraints of the screen's 4 x 6 layout ensure that users are not violating basic HCI principles, such as placing icons too closely together or overlapping them, the problems introduced by inconsiderate placement of icons within the grid persist. For instance, a user is perfectly free to place two nearly identical application icons in proximity on the screen, where they may be confused. Furthermore, a user may also place an icon in a position that is difficult to reach while manipulating the phone with a single hand. Users are generally able to configure the layout within the given constraints without any guidance for satisfying HCI, as well as balance, unity, and sequence.

There are millions of Google results for the query, "Organize apps on iPhone automatically," but these results are limited to help articles on the simple mechanics of moving application icons around, and tips for organizing. Understandably, there's no application programming interface (API) for automatically rearranging application icons on the screen. If there were, this would result in an arms race for preferred positions. The user is on their own when it comes to curating their applications and the layout of the screen.

The first-generation iPhones came with 17 pre-installed applications. Current models come pre-installed with 27 applications. The manufacturer chooses to install its take on the four most commonly used applications in the tray, with the remaining 23 icons occupying all but one of the initially available 24 layout positions on the first screen. If not rearranged (or deleted), these pre-installed apps allow for a single user-chosen app to be placed on the home screen. All others are relegated to the second and subsequent screens. We presume that almost all but the most passive users will rearrange their app icons to suit their personal preferences and style. All but a very few users are neither HCI nor design experts, and, as lay people, adopt a casual or lazy approach to screen layout.

Figure 2: Standard Screen Layouts for iPhone 6-8, X and Xr Max

2 METHOD

In this project, the authors used both trained neural networks and good-old-fashioned Artificial Intelligence (GOFAI) to analyze and suggest an optimized screen layout for home screen captures from user input. The system uses image segmentation based on standard iPhone screen characteristics, a trained classifier and heuristics in order to generate a recommended screen layout and an optimization score for the input image.

2.1 Input Method

Users are asked to use the screen capture function of their phones to generate an image of their home screen via a Google Documents survey. The shared screen captures are uploaded to a shared Google Drive, where they can be input into the web framework for Layout Land.

The web back end enforces a data structure which includes a directory of the user's name, a folder for the submitted screen capture, and a placeholder folder for the image segments that will be used for further analysis.

For the user interface pipeline we used Flask to run a server and the python scripts. First the images are cropped by determining what type of phone it is and the predetermined app positions. Then a histogram script runs on each app image to create a histogram of their colors, followed by our classifier determining what icon class they are. Then the front end uses simple drag and drop Javascript to allow the user to rearrange their apps in a new layout to be scored. (I could make a pipeline figure to include here) (I can add more as I build the gamification)

https://apps.apple.com/us/story/id1484100916

2.2 Neural Net Training

There are approximately twelve broad categories for machine learning algorithms[1]:

- Regression
- Instance-based
- Regularization
- Decision Tree
- Bayesian
- Clustering
- Association Rule Learning
- Artificial Neural Network
- Deep Learning
- Dimensionality Reduction
- Ensemble
- Specialty Approaches (e.g. Feature Selection, Natural Language Processing, Reinforcement Learning)

It is important to choose wisely among these algorithms, as there are varying degrees of suitability for any given purpose, and even with a pre-trained model, as is the case with this project, understanding the underlying principles is important. Since neural networks fail if not properly trained, it is widely accepted practice to reuse a model (a pre-trained model), and then retraining to a given application is the only step necessary. This project relies on an image classifier based on any TensorFlow Hub module that computes image feature vectors computed by Inception V3 trained on ImageNet. For this project, additionally the authors modified an open source project from Test.ai, called Classifier Builder [5] and its underlying pretrained model, MobileNet. MobileNet is a small, low-power module that runs very quickly for a wide variety of uses. MobileNets, in general, are optimized to run even on mobile devices (hence the name), and they compare favorably to larger image classifiers, such as Inception[3]

The model is trained using the ImageNet image library of over 10 million labeled images. For the Inception V3 model, the image sets are composed of approximately 1.3 million images, split into training and evaluation data sets, with 1.28M of the images in the training set. It is not necessary to train the model repeatedly for each application, so it should be noted that we only offer an explanation of the model's initial training. It was only necessary to retrain this general model for the specific use-case of recognizing iPhone app icons.

2.3 Retraining

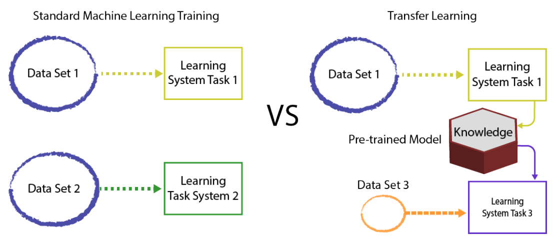


Figure 3: Training versus Transfer Learning

In this project, the authors chose to retrain a model using a subset of predefined iconography, available from the TestAI website.

2.4 Screen Capture Image Segmentation

The predictability of iOS screen dimensions and subsequent layouts allows for the segmentation of the image based on regular measurements. Different techniques, such as edge detection or feature extraction yielded results that were too irregular to be used without intervention.

A screen capture is viewed as a pixel layout. A bounding box corresponding to the dimensions of the icon is drawn. The bounding box is visually moved icon by icon, row by row so that the bounding box encompasses the icon with pixel accuracy. The tool outputs navigation information that is transcribed.

3 EVALUATION

A multi-step evaluation process ensures that classification and heuristics work together to make recommendations on optimized layouts. Some randomness ensures that users can play against the system and some variability in layout results, ensures that users can play multiple times without repetitive results. The following sections describe how the system evaluates input and provide results. The frame size for iPhone 6-8 is 180 x 180 pixels. Pixel Position is calculated from the upper left corner. Delta is calculated from the position of the first frame (-81, -216 for absolute change from 0,0 pixel), therefore horizontal movement can be calculated at +261 pixels per step and vertical movement is +306 pixels per step.

- Position A1 X=81, Y=216
- Position A2 X=342, Y=216 Δ1 (261, 0)
- Position A3 X=603, Y=216 Δ2 (522, 0)
- Position A4 X=864, Y=216 Δ3 (783, 0)
- Position B1 X=81, Y=522 Δ4 (0, 306)
- Position B2 X=342, Y=522 Δ5 (261, 306)
- Position B3 X=603, Y=522 Δ6 (522, 306)
- Position B4 X=864, Y=522 Δ7 (783, 306)
- ...and so on

3.1 Heuristics

It may be possible in the future to develop heuristics based on an analysis of users' submitted layouts. This presumes that people arrange their icons in a generally sensible way, on the whole. But given the overwhelming number of the possible layouts, (millions of possible apps), the required sample size is impractical at this time. Therefore, the authors have chosen to use simple rules to suggest possible, optimal layouts based on a much smaller sample size.

3.1.1 Color Histogram for Finding Dominant Colors. The first heuristics-based rule is to separate icons of similar color. To do this, a simple histogram is extracted for each icon which has been segmented from the screen capture. Most icon designs have purposefully been designed with a dominant color, so it is sufficient to extract the dominant color from the image in order to color sort the icons in the final, suggested layout. The icons are then arranged so that similar color icons are not co-located.

3.1.2 iOS Default Layout. Some users may not deviate from the default layout, but it is expected that a majority of them will. We can presume that many will retain the default applications. We list the default applications, as a baseline for understanding the initial state to place high-value icons in the lower rows. Like the "reachability" function of iOS, this will allow for access to the most frequently used applications, but without requiring the added iOS accessibility feature.

We know from the "reachability" function that the lower portion of the screen is preferred when operating the device with just one hand. For larger devices, we know that the right side of the screen (for the 90% right-handed population) is also preferred, as is anything in the tray, as those icons persist.

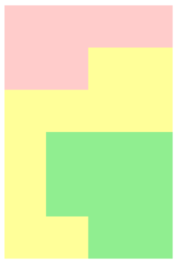


Figure 5: Zones are for reaching applications using one hand are green through red, based on ease-of-use

3.2 Classification

Using the Test.ai Inception V3 classifier, trained against the ImageNet library yielded fairly robust classifications for common icon elements, such as arrows, clouds, clocks, etc. But it did very poorly against less common design elements, such as distinctive fonts, shapes and iconography. It was necessary to collect and train against popular app icons to ensure proper classification of submitted iPhone screen captures. The following application icons were classified correctly, with the classification and confidence percentage.

- YouTube 0.954678
- Instagram - 0.992927
- TikTok - 0.907040
- Gmail - 0.938425
- Facebook - 0.922909
- Google Maps - 0.979775
- Spotify - 0.874147
- Apple Maps 0.293337
- Mail: 0.998767
- Camera: 0.866358
- FaceTime: - 0.646251
- Music: - 0.990503
- Weather: - 0.809097
- Calculator - 0.883121
- Clock 0.380827
- Slack - 0.608549
- Airbnb - 0.913247
- Pinterest - 0.509440
- Apple Calendar - 0.175579

The following are app icons that failed to classify based on the current training.

- AppStore
- Settings
- Reminders
- Health
- Podcasts
- Notes
- Photos
- Phone
- Safari
- Messages
- DoorDash
- WhatsApp

of the layout. The applications are loaded onto the default home screen as indicated (A1...T4). Applications not initially placed on the home screen are stacked on the secondary and tertiary screens. Some applications can be deleted, so users can customize which default apps are installed and where they appear. The following is a list of hone screen, default iOS applications (with placement).

- App Store (D1)
- Books (D3)
- Calendar (A2)
- Camera (A4)
- Clock (B2)
- Health (D4)
- Home (B4)
- iTunes Store (D2)
- Mail (A1)
- Maps (B1)
- Messages (T3)
- Music (T4)
- Notes (C1)
- Phone (T1)
- Photos (A3)
- Reminders (C3)
- Safari (T2)
- Settings (E2)
- Stocks (C2)
- TV (C4)
- Wallet (E1)
- Weather (B3)

Based on the default layout, we can assume that the apps which appear on the home screen are identified by Apple as the most popular. It is difficult to assess whether these default applications are more widely used than third-party applications, but Apple is considering allowing the default applications, such as Safari and Mail, to be replaced with competing applications, which do appear in the top downloaded apps lists, as of March 2020. They include[7]:

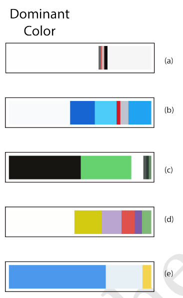


Figure 4: (a) Calendar (b) Safari (c) Spotify (d) Photos (e) Weather color bar histograms

- TikTok
- YouTube
- Instagram
- Snapchat
- Messenger
- Facebook
- Netflix
- Gmail

As tastes change (how many make phone calls on their devices anymore?), the list of preferred apps will change. We would expect that the classification model will need additional, targeted training, based on users' changes in preferred apps.

3.1.3 Reachability. With the release of iPhone 6 in 2014, iPhones have moved away from smaller form factors. This has made single-hand operation more difficult. To address this issue, starting with iPhone 6, the software allows users to enable "reachability mode." When enabled, this allows users to move the top four rows of icons to the bottom, so that they are accessible using just one hand. A1 becomes C1; B1 becomes D1, and so on. It is therefore safe to assume that locating icons in this area (without the use of the "reachability" function) is preferred for one-handed use. Our second heuristic is

- FaceApp
- Uber
- YOLO
- Hulu
- Venmo
- Bitmoji
- Google Chrome
- Amazon
- Netflix
- Messenger
- Snapchat

However, even though these icons failed to classify correctly based on the current retraining of the model, they are still subject to the heuristic evaluation, and should be placed relative to their visual characteristics in optimized locations.

3.3 Optimized Placement

Our method for determining an ideal layout of the user's apps uses a Wave Function Collapse (WFC) algorithm. Using the classifications of apps, popularity ranking, the preferred app locations on the phone screen (reachability), and the histogram data, we create a set of rules that determines where apps should be placed and which apps should not be placed next to each other. With the similarity to WFC, we fill a map of appSpots with a list of every possible app that could be placed there. Then we prioritize choosing "popular" apps for the easy to access app locations. When we have randomly selected an app for a location, we visit its neighbors in the layout map to remove any forbidden neighbors. We also remove the chosen app from every list in the map. Then if we reach a point where the layout is unsolvable, we attempt again, with a finite number of tries. The number of tries is randomized, as each time the WFC is seeded, the optimal positions are shuffled into the worst and the neutral positions, resulting in a variable number of tries and slightly different constructions. All possible constructions, though varying slightly, will conform to the heuristic and reachability rules.

4 GAMIFICATION

Layout Land automatically classifies and makes determinations on optimized layouts for users' home screens, but it cannot rearrange those screens for the users on their devices. The system presents the optimal layout as a matrix of the user's application icons presented as a single image. In order to create an incentive for users to adopt the suggested layout, the system presents them with a simple tile game, where they must move a tile containing the segmented icon from their home screen into a new position. The objective is for the user to guess at a more optimal layout. The user-selected optimization is then compared to the system-selected optimal layout, and a score is returned to the user. The user can then try again to improve their score, or reveal the system's suggested layout.

Since the determination of our optimum layout is based on zones and not exact placement, we score users based on how many apps they placed within a zone of reachability and if they successfully segregated similar colors. The users are not provided with the heuristics or the WFC algorithm goals, so they're placement will be biased by both their preferences (which should coincide with our internal heuristics), and by knowing it's a game. But by scoring when a popular app is placed within a zone, and through color sorting, we expect users to score well. Users generally scored in the XX percentile on their first try without knowing the rules. Users are free to play multiple times, and we would expect scores to improve after the first round, when the rules are revealed with the user score.

5 FUTURE WORK

Given the short duration of development for Layout Land, there were limits to the amount of work that could be done. The following sections outline future work.

5.0.1 Collecting User Data for Additional Heuristics. Currently, no data on the placement of the users' icons is collected. With a sufficient sample size, these data could be used to formulate another heuristic, presuming that people lay out their screens to accommodate their uses in a way that makes sense for them. In the aggregate, these layouts could be used to develop user profiles that could be used to tune optimization suggestions.

5.0.2 Personality Profiling. In order to fully understand the utility of Layout Land and how it relates to the usability of personal mobile devices, the authors should conduct a user study. There has been some research into personal information management (PIM) styles that describe the difficulties of developing general tools for PIM because of variances in how individuals structure and retrieve personal information. Some cues are available based on personality traits based on the study outlined in the paper, "PIM and personality: What do our personal file systems say about us?[4]" We would expect unintentional identity traits (personality) to be evidenced by how users arrange their apps. This would require that users provide personal profiles, likely through an online survey accompanying the screen submission process. Perhaps even more than how people use PIM, the choice and placement of app icons may prove to be strong indicators of personality.

6 CONCLUSION

How one chooses to place icons on a personal mobile device is both highly personal and potentially poorly curated. This project attempts to provide some guidance to users for optimizing their own layouts based on knowledge of the applications they have installed, their placement within the constrained context of the iPhone screen, common HCI factors and heuristics derived from computational methods. We believe this is a sensible framework for better curation and layout of personal mobile devices.

REFERENCES

[1] [n.d.]. ([n. d.]).

[2] Ahamed Altaboli and Yingzi Lin. 2011. Investigating effects of screen layout elements on interface and screen design aesthetics. *Advances in Human-Computer Interaction* 2011 (2011).

[3] Sandler et al. 2020. MobileNet. (March 2020).

[4] Charlotte Massey, Sean TenBrook, Chaconne Tatum, and Steve Whittaker. 2014. PIM and personality: What do our personal file systems say about us? *Conference on Human Factors in Computing Systems - Proceedings* (04 2014). https://doi.org/10.1145/2556288.2557023

[5] Chris Navrides. 2020. testdotai: classifier-builder. (March 2020).

[6] Jason Pascoe, Nick Ryan, and David Morse. 2000. Using while moving: HCI issues in fieldwork environments. *ACM Transactions on Computer-Human Interaction (TOCHI)* 7, 3 (2000), 417–437.

[7] SimilarWeb.com. [n.d.]. The Most Popular iPhone Apps Ranking in United States according to App Store. ([n. d.]). https://github.com/testdotai/classifier-builder