Computational Approaches for App-to-App Retrieval and Design Consistency Check

Seokhyeon Park*[1], Wonjae Kim*[2], Young-Ho Kim[2], and Jinwook Seo[1]

[1] HCI Lab, Department of Computer Science and Engineering, Seoul National University, Seoul, Republic of Korea
[2] Naver AI Lab, Seongnam, Gyeonggi, Republic of Korea *Equal Contribution

ABSTRACT

Extracting semantic representations from mobile user interfaces (UI) and using the representations for designers' decision-making processes have shown the potential to be effective computational design support tools. Current approaches rely on machine learning models trained on small-sized mobile UI datasets to extract semantic vectors and use screenshot-to-screenshot comparison to retrieve similar-looking UIs given query screen- shots. However, the usability of these methods is limited because they are often not open-sourced and have complex training pipelines for practitioners to follow, and are unable to perform screenshot set-to-set (i.e., app-to-app) retrieval. To this end, we (1) employ visual models trained with large web-scale images and test whether they could extract a UI representation in a zero-shot way and outperform existing specialized models, and (2) use mathematically founded methods to enable app-to-app retrieval and design consistency analysis. Our experiments show that our methods not only improve upon previous retrieval models but also enable multiple new applications.

MOTIVATION

UI SEMANTIC REPRESENTATION

EXISTING APPROACHES

Trained with
Small-sized UI Dataset
including UI metadata (e.g., View Hierarchy...)
→ **Specialized UI Models** →
Need to be trained with UI screenshot & metadata
UI Semantic Representation
The distance between vectors indicates the similarity between the screenshots

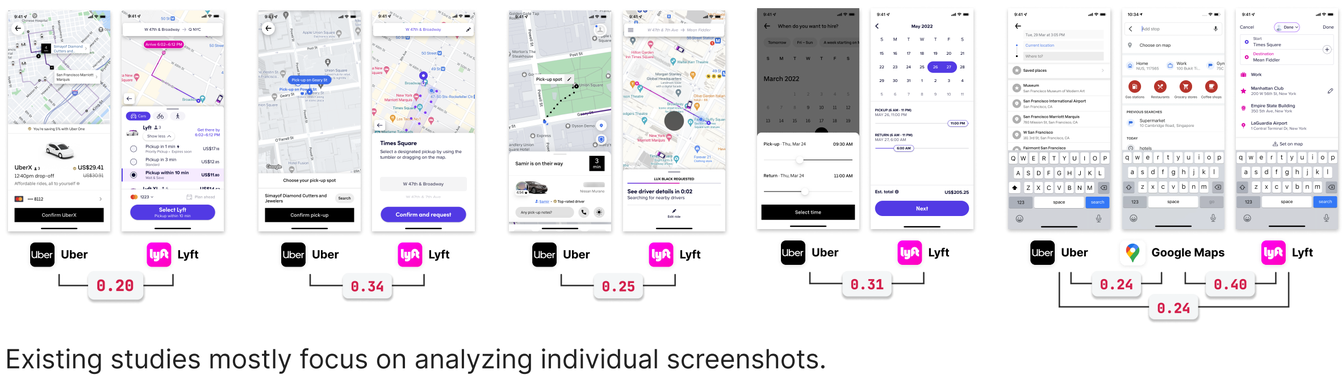
OUR APPROACH

Trained with
Large web-scale images
→ **Foundation Model** →
No need to be trained again with UI data → **Training-free**
UI Semantic Representation

Using Dataset from Mobbin **+ Rico for evaluation**

320 apps, 783 sets of screenshots	9.3k apps, 66k screenshots
Avg. 126 screenshots per app	Avg. < 10 screenshots per app

APPLICATION-LEVEL ANALYSIS



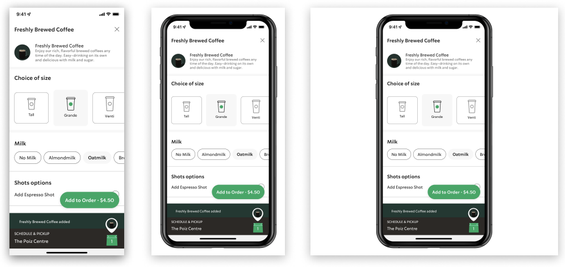
Existing studies mostly focus on analyzing individual screenshots.
Screenshot-based analysis has limitations in considering the **problem domain and functionality of the app**.

APPROACHES

SUPREMACY OF FOUNDATION MODEL

Extract semantic features from a screenshot using OpenAI CLIP image encoder.

Also, inspired by images generated from screen-related text prompt, we augmented screenshots.



Original **Mockup** **Mockup and Squared**

Zero-shot accuracy
app category classification

	Top-1	Top-5
No augmentation	37.68	70.95
+ Mockup	39.67	74.28
+ Mockup + Square	**40.49**	**74.65**

Screen Similarity
N=1250, MTurk

	Score
Screen2Vec	2.92 ± 1.36
TextOnly	3.22 ± 1.30
LayoutOnly	3.00 ± 1.33
Ours	3.50 ± 1.16
Ours + Aug	**3.57 ± 1.08**

OPTIMAL TRANSPORT

$$\mathcal{D}_{ot}(\mathbf{a},\mathbf{b}) = \min_{\mathbf{T}\in\Pi(\mathbf{n},\mathbf{m})} \sum_{i=1}^{n_a}\sum_{j=1}^{n_b} \mathbf{T}_{ij}\cdot c(\mathbf{a}_i,\mathbf{b}_j)$$

$\mathbf{T} \in \mathbb{R}_{+}^{n_a \times n_b}$
Transportation Plan

$c(\cdot,\cdot)$
Cosine Distance

UNIFORMITY LOSS

$$L_u(I_e) \triangleq \log \frac{1}{n^2}\sum_{i,j} G_t(I_e^{(i)}, I_e^{(j)})$$
$$= \log \frac{1}{n^2}\sum_{i,j} e^{2t\cdot I_e^{(i)\top} I_e^{(j)} - 2t}, \quad t>0$$

$I_e \in \mathbb{R}^{n\times d}$
Embedding Vectors

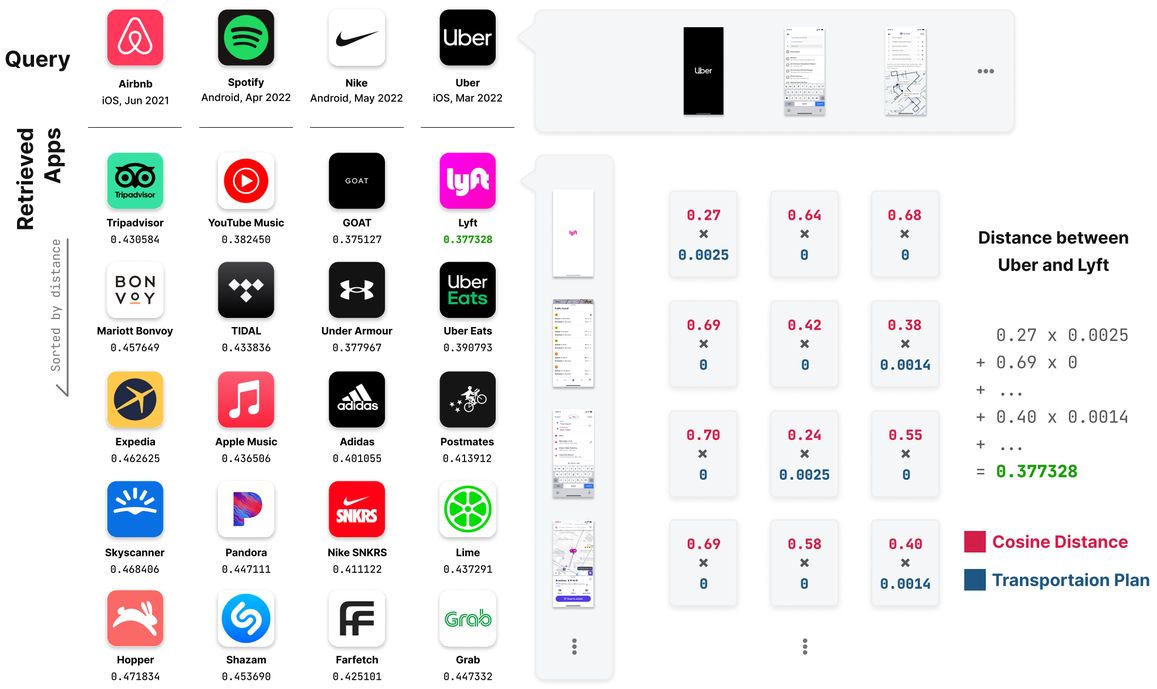
G_t
Gaussian Potential

APP-TO-APP RETRIEVAL

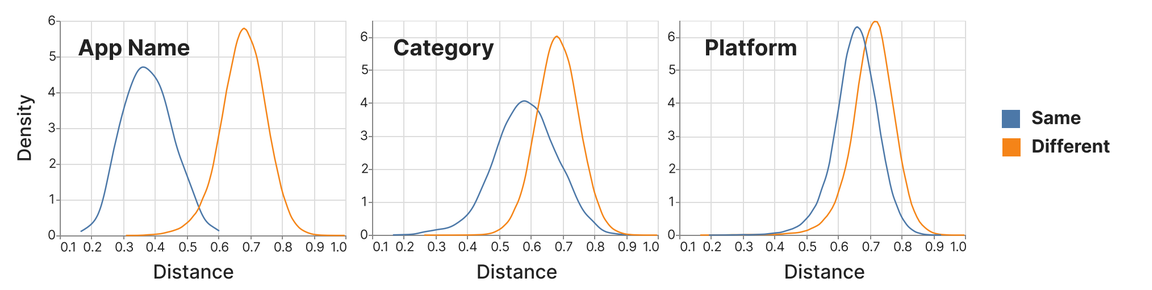
Measure the distance between apps using Optimal Transport

See how retrieved apps are similar to their query apps.

We can analyze further which screens within the two apps are similar by using the transportation plan.



ANALYSIS

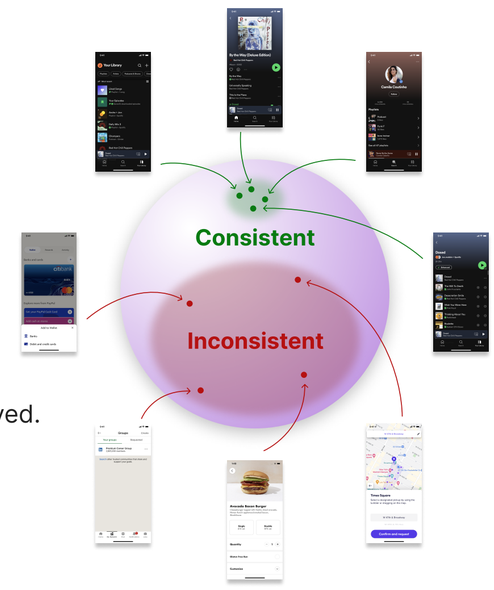


Our dataset consists of apps with multiple versions (release date and platform), and category metadata.
By measuring the pairwise OT distance of apps, we found that **apps in the same group are indeed clustered.**

DESIGN CONSISTENCY CHECK

As CLIP embeds the image onto the **unit hypersphere**, we consider using **Uniformity Loss** as a metric for **design consistency check**.

$L_u \propto$ Design Consistency $\quad -4 \le L_u \le 0\ (t=2)$



EXPERIMENT

Random Change
Replace N images in each app with **randomized images from other apps**.

Within App Change
Replace N images in each app with images **from the same app** that have been reserved.