WHAT IS IT IN A NAME? AN EMPIRICAL EXAMINATION OF BRAND IMITATION IN MOBILE APP MARKET

Mei Li, School of Computing, National University of Singapore, Singapore, limei@comp.nus.edu.sg
Khim-Yong Goh, School of Computing, National University of Singapore, Singapore, gohky@comp.nus.edu.sg
Huseyin Cavusoglu, Naveen Jindal School of Management, University of Texas at Dallas, Richardson, TX, USA, huseyin@utdallas.edu

Abstract

While millions of users feel pity for the removal of Flappy Bird, many Flappy Bird copycats are flowing into the mobile app stores and even rocket up the top rank charts. In the highly competitive mobile app stores, imitating the names of “superstar” apps is a tactic shared by developers. In this paper, we examine the effects of this strategy – how and to what extent it influences the imitator apps’ downloads. To achieve a better understanding, we also investigate the imitation-related factors that affect the imitators’ downloads. With a dataset from Google Play, we evaluate the research questions with propensity score matching and linear regression. The results suggest that, on average, imitating the names of “superstar” apps brings imitators an improvement in downloads. This positive effect is influenced by the number of “superstar” apps that the imitator’s name looks like, the name similarity level, functional similarity and price difference between the imitator and “superstar” apps. Both theoretical and practical implications are discussed.

Keywords: superstar, brand name imitation, name similarity, mobile app stores, app developers, copycat.
1 INTRODUCTION

The removal of Flappy Bird (Wikipedia 2014) from the Apple App Store and Google Play in February may disappoint millions of users who wanted to play the game but had not installed it yet. While these users still feel pity for it, many Flappy Bird copycats are flowing into the app stores and even rocket up the in-store charts (DailyMirror 2014; LosAngelesTimes 2014). Confronted with so many copycat apps, one may want to ask “whether these clones really benefit from the halo of Flappy Bird.

Nowadays, electronic exchanges (e.g., eBay, Apple Store) provide sellers a convenient and concentrated platform to sell their products. Faced with thousands of or even millions of products on the platforms, consumers often resort to search engines to seek their intended goods. They are likely to use the names of “superstar” products as keywords, which consumers get to know from mass media or word-of-mouth. This situation raises a tough question for non-“superstar”, less well-known product sellers – how can their products obtain more visibility and consumers’ share of wallet?

An easy way frequently used by sellers is to imitate or copy “superstar” products’ brand names either in part or in whole, just like the case of Flappy Bird. Such a tactic exploits potential consumer inattention since consumers may mistakenly perceive an imitator as a “superstar” product. This issue has been extensively investigated in the trademark literature, since highly similar brand names sometimes infringe the legal rights of the original brand (Howard et al. 2000; Loken et al. 1986). In addition, a crowded competitive market impacts on not only the sellers, but also the consumers. Consumers cannot easily find products of interest, especially those that require searching and browsing online. Even if consumers did not mistake their products as “superstars”, brand name imitators still hope to gain more exposure along with the “superstars” (e.g., through co-display in the search results). In such a context, how consumers treat the products from the imitators is an open question. Do consumers attribute a low valuation to products from the imitators? Are consumers willing to sample or purchase from the imitators? Does opportunistic brand name copying or imitation pay off for sellers?

In this study, we investigate these questions by empirically examining Google Android’s official app store Google Play, which is one of the most established mobile app platforms. Google Play is known for being a tremendously crowded market with a large supply of mobile apps. If apps cannot obtain enough mobile users’ attention during their debut on the “What’s New” chart, they may hardly get significant visibility again (Sangaralingam et al. 2012). A common tactic that developers usually use to deal with this sobering fact is to copy or imitate “superstar” apps’ names, or even the content to increase the incidences of product offerings co-display (Qiu et al. 2011). Therefore, our research questions specifically are:

(1) How and to what extent does the “superstar” name copying or imitation strategy influence the downloads of the imitator apps in Google Play?

(2) What are the imitation-related factors that may affect the downloads of the imitator apps?

We use the app information (e.g., app name, downloads range) in Google Play from May 2011 to March 2012 to evaluate the research questions. Propensity score matching (PSM) and a linear regression have been employed to obtain some preliminary results of the name imitation phenomenon. The preliminary results show a positive average downloads improvement for brand name copiers or imitators. This positive effect is influenced by the number of “superstar” apps that the imitator’s name looks like, the name similarity level, functional similarity and price difference between the imitator and “superstar” apps.

The remainder of this paper is organized as follows. Section 2 discusses existing studies about brand name imitation. Section 3 introduces our research dataset and methodology. Preliminary results are shown in Section 4. Section 5 concludes the paper with discussion about our contribution and the direction for future work.
2 RELATED STUDIES

Brand is viewed as an identifier of products, which is frequently associated with products’ marketing mix, such as the price, the promotion and the service of a product (Wood 2000). Brand name is a valuable asset for firms since a well-established brand adds value to products. Brand imitation is proliferating nowadays since the imitators can take advantage of the established brands to increase the value of their own products. There is a long literature about brand imitators in marketing and economics. By naming or dressing themselves like well-known brands, imitators make consumers confused about the origin of imitators and the imitated brand. When the confusion level is high, consumers may naturally associate the positive evaluation of the original established brand with the imitators. Consequently, highly similar imitators oftentimes can receive higher valuation from consumers (Loken et al. 1986; Warlop & Alba 2004).

However, several studies raised different opinions by contending that this relationship is moderated by some contingent factors, such as degree of consumer care and imitation type. Howard et al. (2000) find that consumers’ brand confusion is related to the degree of consumer care as well as the similarity type between products. When the degree of consumer involvement is high, generally a higher level of brand confusion will be resulted in. When the degree of consumer care is low, only the similarities that can be easily processed (e.g., sound-based similarity) lead to greater information confusion level. The study of d’Astous and Gargouri (2001) suggests that the image of the stores, where the products are distributed, positively relates to consumers’ evaluation of luxurious brand imitators. However, for convenience goods, this effect disappears when the imitated brand is present with the brand imitators. Van Horen and Pieters (2012a) examine the influence of imitation types (feature imitation vs. theme imitation) on consumers’ evaluation of the brand imitator. They find that compared to feature imitation, which may cause consumers’ resistance, theme imitation is more acceptable. Another study by Van Horen and Pieters (2012b) investigates the impact of similarity level on customers’ evaluation of the brand imitator with the presence of the original brand. The results suggest that blatant copycat is more perilous than subtle copycat. Both d’Astous and Gargouri (2001) and Van Horen and Pieters (2012b) find that consumers lower their valuation of imitator brands with the presence of the original established brands. In this scenario, juxtaposition of imitators and original brands induces consumers into a contrastive thinking mode. Consumers may conclude that imitators are trying to leverage on the original established brand’s good image to promote themselves. As a result, comparative evaluation takes the positive association between the imitator and its original brand away.

Almost all the prior studies on brand imitation require participants to make decision among limited number of brands due to their experimental nature. However, in reality, especially online, consumers face many alternative choices, such as over half million apps in Google Play. As attention becomes a scarce resource, online consumers bear a relatively high search cost (Falkinger 2008). Under such circumstance, how do consumers react to brand imitation? Would they patronize brand imitators even with the presence of the original brand, because co-appearance greatly saves their search effort? Above all, consumers’ reaction to brand imitators in an information overloaded context is still unsolved, and this issue is what we are trying to address in this study.

3 DATA AND METHODOLOGY

We obtained app information on Android platform from Mobilewalla (Datta et al. 2011), a website that keeps tracking app information from Android’s official store, Google Play, since the middle of May 2011. The latest observation in this dataset was on March 26, 2012. The data contained each app’s basic information (e.g., name, developer, category, size, and downloads range), version updating history and in-store rankings.

We identified imitators by comparing the similarity of a particular app’s name and “superstar” apps’ names. We first obtained a “superstar” list comprising 659 apps which had been ranked in the top 100 popular chart at least once. Then, Levenshtein distance (Levenshtein 1966) was used to quantify the similarity of remaining apps’ name and these “superstar” apps’ names. Levenshtein distance is
defined as the minimum addition, deletion or substitution of a single character needed to transform one sequence to the other. Since there were 344,823 apps in our dataset, pairwise Levenshtein distance computation would have required a tremendous computational capacity. Therefore, we narrowed down the comparison pool by selecting those app names which, at least, had one common word with the “superstar” apps’ names. This process resulted in 102,083 name pairs. We kept top 25 similar app names for each “superstar” app according to Levenshtein distance. If two apps had the same Levenshtein distances, the one sharing more common words with the “superstar” app’s name got higher priority. Finally, we ended up with 15,146 pairs, consisting of 659 distinct “superstar” apps and 12,569 distinct imitator apps. Among them, 204 apps were both “superstars” and imitators. Since popularity may beget further popularity (Carare 2012), distinguishing the downloads improvement effects of being ranked top on app downloads from the effects of having similar name with “superstar” apps is difficult. To remove the impact of being ranked on top charts, we dropped these 204 apps from our analysis.

In order to estimate the effects of having similar names with “superstars” on imitators’ performance, we should know the imitators’ performance if they are not imitators. However, this is counterfactual and we can never observe. Having a similar name with a “superstar” actually is a treatment. A compromising strategy is to use PSM to seek the counterparts which should have had this treatment but did not (Caliendo & Kopeinig 2008; Heckman et al. 1997). Then, we can compare the performance of apps with and without treatment to get the treatment effect. Since Google Play does not have detailed downloads count but provides downloads range for each app, we use this ordinal variable to measure apps’ performance.

To conduct PSM analysis, we need to identify a set of observable covariates X that influences treatment decision and apps’ performance simultaneously. The goal is to balance the distribution of covariates and make the difference of outcome attributable to the treatment only. The selection criteria for X include (1) they affect both treatment decision and outcome; (2) they are unaffected by treatment decision or anticipation of it (Caliendo & Kopeinig 2008; Morgan & Harding 2006). Having a similar name with a “superstar” is highly coincident among the enormous number of potential apps and difficult to predict. Instead of predicting selection into this treatment as well as possible, the main purpose of PSM is to balance all covariates (Caliendo & Kopeinig 2008). Thus, we are more concerned about seeking for an X vector which greatly influences the outcome (i.e., apps’ performance).

A tactic commonly utilized by non-brand name app developers to market their apps is imitating “superstar” apps (Qiu et al. 2011). By using a similar name or even copying “superstar” apps’ content, imitators can obtain more exposure rapidly and attract downloads from curious and sometimes careless consumers. However, for well-known developers, they are less likely to knockoff others’ works. Instead, they concentrate on establishing and enhancing their brand image by differentiating themselves from their competitors. Therefore, we posit that developers’ previous experience of publishing influences their decision to imitate “superstar” apps. We use multiple factors to measure developers’ previous experience of publishing (definitions are given in Table 1). As the time span of our dataset starts from May 2011, we are unable to observe the whole publishing history of developers who entered into Android before this time point. Hence, only those apps whose developer entered into the platform after May 2011 were kept.

To balance the effects of covariates on the outcome of treated and non-treated groups, we also need app-specific characteristics that may affect apps’ downloads (definitions are given in Table 1). The age of an app affects total downloads, since downloads is a non-decreasing function of the age. Though most mobile apps cost little, the price does affect users’ evaluation of them. Developers often promote their apps with price discount. To capture this promotion effect, we incorporate the standard deviation of price as an indicator that measures the apps’ promotion intensity. Apart from price, user rating is also a crucial factor since highly rated apps may be able to attract more downloads due to word-of-mouth effects. The number of app versions controls for developer’s update and maintenance frequency. Apps updated frequently are more likely to be downloaded. As entertainment goods, mobile apps experience great holiday and weekend effects. Thus, we include Christmas and time dummies into the propensity score estimation. In addition, apps in the same category might receive
similar exogenous shocks to their sales and we also account for category-specific effects by including category dummies.

<table>
<thead>
<tr>
<th>Covariate</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DevAge</td>
<td>total weeks passed since the developer published his/her first app</td>
</tr>
<tr>
<td>PreApp</td>
<td>total number of apps the developer published before</td>
</tr>
<tr>
<td>PreFreeApp</td>
<td>total number of free apps the developer published before</td>
</tr>
<tr>
<td>PreHitApp</td>
<td>total number of hit apps (i.e., ranked in top 500) the developer published before</td>
</tr>
<tr>
<td>PreVersion*</td>
<td>average number of versions previous apps have</td>
</tr>
<tr>
<td>PreRatingCnt*</td>
<td>average rating count of previous apps</td>
</tr>
<tr>
<td>PreRatingScore*</td>
<td>average rating score of previous apps</td>
</tr>
<tr>
<td>PrePromotion*</td>
<td>average price standard deviation of previous apps</td>
</tr>
<tr>
<td>PreCategory*</td>
<td>number of different categories the developer has published apps before</td>
</tr>
<tr>
<td>AppAge</td>
<td>total weeks passed since the app was being published</td>
</tr>
<tr>
<td>AppPrice</td>
<td>price of the app</td>
</tr>
<tr>
<td>AppPromotion</td>
<td>standard deviation of the app’s price</td>
</tr>
<tr>
<td>AppRatingScore</td>
<td>app’s consumer rating score</td>
</tr>
<tr>
<td>AppVersion</td>
<td>total versions the app has</td>
</tr>
<tr>
<td>ChristmasDummy</td>
<td>1 if the app was published in two weeks before or after Christmas; 0 otherwise</td>
</tr>
<tr>
<td>TimeDummies</td>
<td>dummies for each day of a week, in order to capture weekend effects</td>
</tr>
<tr>
<td>CategoryDummies</td>
<td>dummies for different categories, in order to capture category-specific effects</td>
</tr>
</tbody>
</table>

Notes: * if focal app is the developer’s first app, the value of those variables is 0.

Table 1 Definition of Covariates

4 PRELIMINARY RESULTS

We employed psmatch2 (Leuven & Sianesi 2003) in Stata 12.0 to conduct PSM analysis. Our final dataset consists of 104,576 apps, of which 3,671 are treated (imitators) and 100,905 are untreated. The estimation of propensity score was done with a Logit regression, which is shown in Equations (1), (2) and (3). The results of Logit estimation are reported in Table 2.

\[ Imitator_i^* = X_i \beta + \epsilon_i \]  
\[ Imitator_i = 1 \text{ if } Imitator_i^* > 0, \text{ and } Imitator_i = 0 \text{ otherwise} \]  
\[ \text{Prob}(Imitator_i = 1 | X_i) = \frac{\exp(X_i \beta)}{1+\exp(X_i \beta)}, \]  
\[ \text{Prob}(Imitator_i = 0 | X_i) = \frac{1}{1+\exp(X_i \beta)} \]  

where \( i \) denotes imitator app and \( X_i \) is a vector of covariates that determine name imitation propensity as shown in Table 1.
Variable | Coefficient | Variable | Coefficient
---|---|---|---
DevAge | -0.000 | PrePromotion | -0.010
PreApp | -0.001*** | PreCategory | 0.015*
PreFreeApp | -0.004*** | AppAge | -0.023***
PreHitApp | 0.188*** | AppPrice | -0.002
PreVersion | 0.021*** | AppPromotion | 0.014
PreRatingCnt | 0.000 | AppVersion | 0.040***
PreRatingScore | -0.045*** | AppRatingScore | 0.044***

Notes: 1. Dependent variable: imitator.
2. Observations = 104,576, pseudo-R² = 0.039. *** p<0.01, ** p<0.05, * p<0.1.
3. Christmas, time and category dummies are included but omitted here.

Table 2 Estimation Results of Logit Model

We acquire average treatment effect on the treated (ATT) with five matching estimator (i.e., 1 nearest neighbor, 3 nearest neighbors, Caliper with tolerance level 0.01, Caliper with tolerance level 0.05 and kernel). Table 3 shows the matching results. All the five matching estimators suggest a significant positive ATT. It means that having similar names with “superstar” apps does benefit imitators’ downloads. Compared with the non-imitation peers, the imitator apps have higher downloads range.

<table>
<thead>
<tr>
<th>Estimator</th>
<th>Treated</th>
<th>Control</th>
<th>Effect</th>
<th>Std. Err.</th>
<th>T-value</th>
<th>OffSup</th>
<th>BiasBef</th>
<th>BiasAft</th>
<th>R² After</th>
</tr>
</thead>
<tbody>
<tr>
<td>1NN</td>
<td>6.94</td>
<td>6.57</td>
<td>0.37</td>
<td>0.067</td>
<td>5.56</td>
<td>0</td>
<td>7.50</td>
<td>1.25</td>
<td>0.010</td>
</tr>
<tr>
<td>3NN</td>
<td>6.94</td>
<td>6.55</td>
<td>0.39</td>
<td>0.055</td>
<td>7.09</td>
<td>0</td>
<td>7.50</td>
<td>0.93</td>
<td>0.008</td>
</tr>
<tr>
<td>Caliper(0.01)</td>
<td>6.93</td>
<td>6.56</td>
<td>0.37</td>
<td>0.067</td>
<td>5.57</td>
<td>6</td>
<td>7.50</td>
<td>1.23</td>
<td>0.010</td>
</tr>
<tr>
<td>Caliper(0.05)</td>
<td>6.94</td>
<td>6.56</td>
<td>0.37</td>
<td>0.067</td>
<td>5.57</td>
<td>2</td>
<td>7.50</td>
<td>1.25</td>
<td>0.010</td>
</tr>
<tr>
<td>Kernel</td>
<td>6.94</td>
<td>6.23</td>
<td>0.71</td>
<td>0.048</td>
<td>14.81</td>
<td>2</td>
<td>7.50</td>
<td>5.06</td>
<td>0.028</td>
</tr>
</tbody>
</table>

Note: 1. OffSup is the number of apps outside common support.
2. BiasBef/BiasAft is the mean standardized bias (absolute value) over all IVs before/after matching.

Table 3 Matching Results

Though the imitator apps, on average, have better performance than the apps without imitation, there still exists variation in the imitator apps’ performance. Next question we want to answer is what imitation-related factors influence the imitator apps’ downloads. Several imitation-related characteristics have been incorporated into our econometric model. The number of “superstar” apps the imitator app imitates may influence its visibility. The more “superstar” apps it looks like, the higher probability it will be shown to users. Similarly, the rank of the “superstar” app may impact users’ search interest as well. As higher rank is smaller in its number, it might negatively influence the focal imitator app’s downloads. The Levenshtein distance between the “superstar” app and the imitator app itself decides the imitator app’s display position on the page. The larger the distance, the lower position the imitator app will be co-displayed with the “superstar” app. It might be harder for consumers to notice the apps in lower position. Thus, the Levenshtein distance may negatively impacts the focal imitator app’s downloads.

The price of the “superstar” app that the imitator app imitates may also impact the performance of the focal app since “superstar” app of lower price possibly can attract more user search. The growing number of user search of the “superstar” app is able to increase the visibility of the focal imitator app. Moreover, the price difference between the “superstar” and the imitator app may also influence users’ willingness to download the focal app. If the imitator app is much cheaper than the original “superstar”, users are more likely to download the imitator app.
In addition, our name comparison neglects the functional similarity of the apps. Sometimes the focal imitator app may not be a real imitator; instead, it only acts as a plug-in or assistance tool for the “superstar” app. For example, there are many plug-in apps for the famous Angry Birds to help users continue their plays from their latest failure level, which function is not supported by the game itself. In such case, users may not be affected by the negative influence brought by contrastive thinking mode (d’Astous & Gargouri 2001; Van Horen & Pieters 2012b). Rather, they may welcome these plug-ins to improve their user experience with the “superstar” apps. To control for this, we use the apps’ category information to infer their functional dissimilarity. Specifically, if the “superstar” app and the focal imitator app are in different categories, we treat them as complements.

Apart from the imitation-related factors stated above, the characteristics of imitator app’s developer may influence the performance of the imitator app. We also include them into our econometric model, which is shown in Equation (4). The definitions of the newly added variables can be found in Table 4. As an imitator app may imitate multiple “superstar” apps, all the variables in Table 4 except SuperstarNum are aggregate measures (please refer to Table 4 for details).

\[
\text{DownloadRange}_i = \beta_0 + \beta_{\text{SuperstarNum}_i} + \beta_{\text{SuperstarRank}_i} + \beta_{\text{LevenshteinDistance}_i} + \beta_{\text{SuperstarPrice}_i} + \beta_{\text{PriceDif}_i} + \beta_{\text{Complement}_i} + \gamma X_i + \epsilon_i
\]

where \( i \) denotes imitator app and \( X_i \) is a vector that denotes the variables listed in Table 1.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SuperstarNum(_i)</td>
<td>total number of “superstar” apps that app ( i ) imitates</td>
</tr>
<tr>
<td>SuperstarRank(_i)</td>
<td>average rank of “superstar” apps that app ( i ) imitates</td>
</tr>
<tr>
<td>LevenshteinDistance(_i)</td>
<td>average Levenshtein distance of “superstar” apps that app ( i ) imitates</td>
</tr>
<tr>
<td>SuperstarPrice(_i)</td>
<td>average price of “superstar” apps that app ( i ) imitates</td>
</tr>
<tr>
<td>PriceDif(_i)</td>
<td>price of imitator app ( i ) minus SuperstarPrice(_i)</td>
</tr>
<tr>
<td>Complement(_i)</td>
<td>equals 1 if app ( i ) imitates at least one “superstar” app which is in a different category from ( i ); 0 otherwise</td>
</tr>
</tbody>
</table>

Table 4 Definition of Newly Added Variables

Table 5 shows the results. As we are only interested in the imitation-related factors, which are manageable for developers, the coefficients of developer characteristics have been omitted from the table (available upon request). Almost all the variables in Table 5 significantly influence imitator apps’ performance except for SuperstarRank and AppAge. The positive coefficient of SuperstarNum suggests that the more “superstar” apps’ names the imitator looks like, the higher the downloads range will be. LevenshteinDistance is positively related to downloads range, different from our conjecture. It means that consumers do not prefer highly similar imitators, since LevenshteinDistance is negatively related to the similarity level of the imitator and imitated “superstar” apps’ names. This is consistent with the findings by Van Horen and Pieters (2012b), who suggest that moderately similar imitators can perform better than highly similar ones.

<table>
<thead>
<tr>
<th>Variable</th>
<th>Coefficient</th>
<th>Std. Err.</th>
<th>Variable</th>
<th>Coefficient</th>
<th>Std. Err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>SuperstarNum</td>
<td>0.355**</td>
<td>0.066</td>
<td>Complement</td>
<td>-0.295***</td>
<td>0.084</td>
</tr>
<tr>
<td>SuperstarRank</td>
<td>0.001</td>
<td>0.001</td>
<td>AppAge</td>
<td>-0.002</td>
<td>0.004</td>
</tr>
<tr>
<td>LevenshteinDistance</td>
<td>0.026***</td>
<td>0.010</td>
<td>AppPromotion</td>
<td>0.138***</td>
<td>0.047</td>
</tr>
<tr>
<td>SuperstarPrice</td>
<td>-0.242**</td>
<td>0.012</td>
<td>AppVersion</td>
<td>0.177***</td>
<td>0.014</td>
</tr>
<tr>
<td>PriceDif</td>
<td>-0.217***</td>
<td>0.019</td>
<td>AppRatingScore</td>
<td>0.654***</td>
<td>0.022</td>
</tr>
</tbody>
</table>
Notes: 1. Observations = 3,671, Adj-R² = 0.3783. *** p<0.01, ** p<0.05, * p<0.1.
   2. We did not include imitator app’s price in avoid of multicollinearity.
   3. Developer characteristics have been included but omitted here.

Table 5 Estimation Results of Model (4)

The coefficient of SuperstarPrice indicates that imitating lower-price “superstars” possibly can get more downloads, consistent with our earlier argument. PriceDif has a significant negative coefficient. Since the lower the price of the imitator app, the more negative PriceDif is, the negative coefficient indicates that pricing the imitator under the price of the “superstar” apps increase the imitator’s downloads. Meanwhile, it also suggests that the lower the price of the imitator app, the higher downloads range it can reach.

Binary variable Complement has a negative relationship with downloads range, opposite to our analysis before. It suggests that users are more likely to select imitators which share the same category with their imitated “superstars”. This is somewhat counterintuitive, since complementary products usually can promote each other’s sales. However, in our context, consumers face a super crowded market and finding an interesting app is not easy. They may encounter imitators when they search for the imitated “superstar” app. The imitators that fall in the same category with the “superstar” app just provide alternative choices for users. The coefficient of AppPromotion demonstrates the price sensitive nature of consumers. The last two variables indicate that apps updated frequently and rated highly can attract more downloads.

5 CONCLUSION AND FUTURE WORK

Our study differs from the prior literature in three ways. (1) To the best of our knowledge, it is one of the first studies that investigate brand imitation out of laboratory settings. (2) It examines information goods, whereas previous studies mostly discuss grocery goods. (3) It explores consumers’ reaction to brand imitation under information overload. The preliminary results also provide meaningful implications for practitioners. On average, imitating “superstar” apps brings benefit to imitators’ downloads. However, highly similar copying may discount this positive effect. Thus, developers need to carefully name their apps and design app content to induce consumers’ positive association with the “superstars”. Furthermore, it is beneficial for imitators to have similar function as the “superstars” they imitate because consumers are more likely to include those imitators into their consideration set. In addition, lower-priced “superstar” apps might be able to bring more downloads to their imitators. Imitation practitioners are suggested to price their apps lower than the prices of the imitated apps.

The main challenge we encountered in this study is the matching quality of treated and control groups. Predicting the probability to imitate is really hard, since the use of this tactic is somewhat random among developers. Some developers may feel shamed and opportunistic to imitate while others may consider it a useful trick to market their apps. All of these are relevant to developers’ unobserved moral value. It makes predicting imitation difficult, hence the relatively small Pseudo-R² in our Logit model. Thus, we will seek for more possible matching methods to deal with the unobservable covariates. Moreover, we account for imitation only by comparing similarity of app names. However, consumers might evaluate apps from different aspects, such as icons, descriptions and screenshots. It is better to incorporate these characteristics to gauge apps’ similarity. Finally, downloads range is an ordinal measurement. If we can obtain real download counts, the results would be more robust.

As one of the pioneers to examine the copycat phenomenon in mobile app stores, we have obtained many interesting findings in this study. Yet, further improvement and investigation are needed. (1) Unobserved heterogeneity has not been accounted for in current paper. Though many developer-specific characteristics are unobserved by us, simulation can be conducted with reasonable assumptions. (2) As downloads range is ordinal measurement, ordinal regression can be used as robustness check. (3) To strengthen the validity of our findings about the factors that influence imitator apps’ performance, more relevant literature should be included to provide references. (4) Lacking users’ search log, it is difficult to understand why and under what circumstances consumers choose copycat apps. As an alternative method to investigate this, we may conduct a survey among mobile app users to understand their search habits and attitudes to copycat apps.
REFERENCES


