POPULARITY AND AUDIENCE MEASUREMENT OF MOBILE APPS: ENABLING EFFECTIVE MOBILE ADVERTISEMENT

SANGARALINGAM KAJANAN
Bachelor of Science (First class) Honors in Information Technology

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY
DEPARTMENT OF INFORMATION SYSTEMS
NATIONAL UNIVERSITY OF SINGAPORE
2014

DECLARATION

I do hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

Further, this thesis has not been submitted previously for a degree at any university.

S. Kja

SANGARALINGAM KAJANAN

27th March 2015

ACKNOWLEDGEMENTS

I take this opportunity to extend my wholehearted thanks to all those who have contributed to make this Ph.D. thesis a success. My wholesome gratitude goes to my almighty goddess for her blessings throughout my life and academic advancement, including the research work.

First and foremost, I offer my utmost gratitude to my advisor and guru Associate Professor Anindya Datta who has supported me throughout my Ph.D. career. I dedicate this thesis to him. He has spent his valuable time patiently with me without any limitation and he has molded me with his knowledge, skill and ability. Without his personal attention, this research would not have been accomplished. I have been very fortunate to have him as my Ph.D. advisor who believed me and realized my potential ability. He has motivated me to be the best that I could be. His valuable advice has considerably influenced in my academic experience. To me, Prof. Anindya is not only an academic advisor, but also a role model and a lifetime mentor.

I cannot find words to express my gratitude to Associate Prof Kaushik Dutta for being my co-advisor throughout my Ph.D. career. His expertise and practical approach are founding pillars of my research. His feedbacks were fundamental support in flourishing this research. It was a great privilege and honor to have him as my co-supervisor during the period of my Ph.D. and I am very much thankful to him for being helpful towards me always.

Besides my advisors, I would like to express my sincere gratitude to the members of my thesis committee, including Prof. Teo Hock Hai, A/Prof. Jungpil Hahn, A/Prof Rudy Setiono and external examiners, for their critical readings and constructive criticisms, which have been extremely helpful in refining and enriching my dissertation. I'm greatly benefited from their encouragements, brilliant ideas and high standard clarifications. It is an incredible honor to be examined by such knowledgeable people. My very special thanks go to A/Prof Jungpil and A/Prof Rudy Setiono for their instructive guidance, insightful criticism and raising timely queries.

Faculty members at the external universities have also contributed to the success of my PhD study. Particularly, A/Prof Debra VanderMeer has given me valuable insights on how to sell your work by being as a good writer and A/Prof Narayan Ramasubbu has given me valuable advice on how to be an effective researcher. I would also like to convey my thanks to all faculty members of School of Computing, who have helped me in numerous ways. I am very much grateful to the School of Computing for providing the financial support for my study.

I would also like to deeply thank the Engineering Team of Mobilewalla, specially Mr. Sameer, Mr. Gokul, Ms. Lian and Mr. Swapnil. They never hesitate to provide me with the required data whenever I requested for it. No words to thank them for their patience, knowledge and attitude.

My thanks to my wife, Duwaraka, are beyond words. Her love and guidance kept me strong and motivated to overcome all the difficulties of my academic life. A mere expression of thanks does not suffice for the continuous support of my mother, parents-in-law, brothers - Dr. Ahilan and Mr. Ragulan and in-laws. Their kindness and support warmed my heart and gave me the courage to look forward and progress.

I also gratefully acknowledge the support that I received from the members of staff in the School of Computing – specially Ms. Loo Line Fong, Mr. Bartholomeusz Mark Christopher and other Technical service personnel who made it very easy for me to handle the administrative and technical issues.

Last but not least, I would also like to thank several other friends and lab mates at the School of Computing who helped me in my PhD study in some way or other.

May the Almighty God richly bless all of them.

TABLE OF CONTENTS

Summary

Consumer software applications that run on smartphones (also known as “mobile apps”, or simply, “apps”) represent the fastest growing consumer product segment in recent times, and cumulative app downloads continue to grow at a fast pace. Popular app stores like Google Play and iTunes are leading in this mobile app revolution. Aligned with this spectacular growth of the mobile market in general and mobile apps in particular, the world of digital advertising has also witnessed a pivot role in mobile media. Mobile devices, and apps, offer an opportunity to reach a large (>800MM globally), diverse, global and engaged audience.

This thesis focuses on mobile advertising. In particular, emphasis is given to the most key component of the advertising workflow- media acquisition, popularly recognized as media buying. The onset of digital advertising, on the web, brought about the emergence of a different way to acquire media – without human intervention, through an automated process, commonly referred to as Programmatic Media Buying (PMB). In this dissertation, a conceptual framework is built up based on PMB as a main means of acquiring inventory in mobile apps. To this end, we focus on how programmatic media buying could help in designing effective mobile ad campaigns. For programmatic purchasing of advertising inventory to be widely adopted, a set of open research problems need to be addressed first. This thesis addresses three vital problems which can be used to enable the PMB in mobile advertisement context. Such that, we posit in this thesis

that ad campaigns would be more effective when there is a way to determine the popularity signals in real time and when there is a way to pass the relevant audience information of mobile apps from which ad requests are coming.

The first study of this thesis focuses on how the real time discovery of social media mentions for an app can be performed and it details how it would help enabling the PMB in the context of mobile advertisement. Particularly an interesting issue that we have addressed in this study is, to evaluate the popularity of specific mobile apps by analyzing the social conversation on them. As such, this study presents a strategy to reliably extract twitter posts which are related to specific apps.

The second study of this thesis focuses on computing real time popularity ranks of mobile apps. As the popularity of apps is highly transient, traditional advertisements delivered based on persistent popularities will not hold for mobile apps. As mobile app popularity is highly transient, for mobile app advertisers knowing popularity rank in real time is very vital, because they are interested in placing their advertisements in apps with the greatest reach; As the existing native store ranks (ranks provided by app stores) have often been criticized for being commercially driven and not representing the “true” popularity rank, we propose a new Ordered Weighted Average (OWA) based ranking mechanism - Deviation based OWA (DOWA), which is an adaptive and dynamic weighting scheme, in which the weights attached to various features dynamically change based on

importance of it in the ranking mechanism. The proposed approach is validated using life cycle data of apps in the Apple iOS app store.

The third study of this thesis is focuses on proposing a non-panel based reliable classification based text mining approach to measure mobile app audience. Proposed dynamic approach can be used to estimate the audience of existing 1.5 million apps as well as the incoming new apps.

In summary, this thesis has focused on programmatic buying of social media popularity, popularity rank computation and audience measurement in the context of mobile apps. These strategies can be used in designing effective mobile ad-campaigns.

List of Figures

List of Tables

Chapter 1
Introduction

1.1. Background and Motivation

Mobile apps (or simply apps) represent the fastest growing consumer product segment of the decade. The total number of apps in the app stores and their rate of growth are remarkable. As of May 2013, there were about 1.4 million active apps available in Apple iTunes and Android Google Play app stores[1], and their growth rate is at least 4% on a monthly basis. Flurry reports that on average a US consumer spends 2 hours and 38 minutes per day with smartphones and tablets and out of this time, 80% (2 hours and 7 minutes) is spent inside mobile apps. Besides on average a consumer launches 7.9 apps per day[2]. According to market estimates, in the first quarter of 2013 the four leading app stores (i.e. Apple, Android, Blackberry and Windows) had 13.4 million app downloads and yielded a revenue of $2.2 billion[3]. This lucrative mobile app revenue is expected to reach $38 billion by 2015[4].

Aligned with this spectacular growth of the mobile market in general and mobile apps in particular, the world of digital advertising has also witnessed a pivotal growth in the mobile media. As mobile devices, and apps, offer an opportunity to

[1] https://www.mobilewalla.com/

[2] http://blog.flurry.com/bid/95723/Flurry-Five-Year-Report-It-s-an-App-World-The-Web-Just-Lives-in-It

[3] http://mashable.com/2013/04/08/canalys-report/

[4] http://bits.blogs.nytimes.com/2011/02/28/mobile-app-revenue-to-reach-38-billion-by-2015-report-predicts/

reach a large (>800MM globally), diverse, global and engaged audience the mobile advertisements are regarded as effective. A recent study showed that 40% of males in the US aged from 18 to 29 "somewhat" or "very much" like mobile ads[5]. Further, 66% of the users who claimed to have interacted with advertising in a magazine app, and 40% have agreed to make a purchase as a result. Moreover, we observe that mobile has become the first screen and made TV as the second screen during the recent Superbowl event[6]. While recognizing the growing importance of mobile apps as a fertile medium for serving ads[7], “Berg Insight” reports that the global mobile ad market will grow from $3.4 billion in 2010 to a mammoth $22.0 billion in 2016[8].

Though there is a huge trend for serving the ads in mobile apps, studies have shown that they yield less revenue than expenses. Mobile ad networks claim that while a lot of money rustle into mobile advertising, it hasn’t been effective or flourishing when it comes to Return On Investment (ROI)[9]. Thus ad agencies consider mobile ads are still far from being “A Cash Cow”[10]. Besides some marketing studies report that, despite massive growth in media consumption and time on the smart phone apps, the mobile ad spending is still at very low level

[5] http://www.emarketer.com/Article/How-Millennial-Men-React-Mobile-Ads/1008834

[6] http://blog.flurry.com/bid/93898/The-Screen-Bowl-Mobile-Apps-Take-On-TV

[7] http://www.emediavitals.com/content/mobile-advertising-numbers

[8] http://www.btobonline.com/article/20120312/WEB02/303129948/mobile-marketing-seeks-afoothold

[9] http://www.businessinsider.com/here-is-the-evidence-that-mobile-advertising-is-in-a-bubble-2012-8?IR=T

[10] http://paidcontent.org/2011/10/11/419-pandora-learns-the-hard-way-mobile-ads-are-still-far-from-being-a-cash/

with only 5-10% of the average brand or agency budget[11]. This is evidenced from Nielsen's recent report that stated "though Internet advertising continues to be a growing medium, it remains a small player"[12]. The report indicates that while global display advertising across the web, mobile and apps grew by 32.4% in 2013 by far the biggest leap of any media, still worked out to only 4.5% share of the overall spending in ads. In contrast, television grew only 4.3%, but remains the highest portion when it comes to ad spending, taking nearly 58% of the market.

Above findings indicate that most of ad campaigns planned for mobile devices are not effective and brand owners' yield a very low return over investment in the mobile ad business. Thus the mobile ad spending is very low compared to other media. Despite the widespread popularity, huge media consumption and time and potential to persuade millions of people, it is unfortunate to learn that mobile advertising is not effective and has not yielded expected ROI. Thus, motivated the aim of this thesis is to build a mobile ad framework that can guide the effective mobile ad campaigns and yield an attractive ROI. Further, we propose a scientifically driven mobile ad framework and also provide solutions to various problems faced by constituents of app ecosystem.

[11] http://www.mobilemarketer.com/cms/opinion/columns/16349.html

In order to design effective ad campaigns, real time media acquisition popularly known as *media buying* should happen in mobile advertising based on some metrics. On its basis, advertising is concerned with matching supply or media (e.g., section of a newspaper, TV show, website or mobile app) to demand, or campaigns (e.g., the launch of a sports shoe for women). The key goals, in executing virtually all advertising campaigns are twofold: given a campaign, (a) acquire the "optimal" media (i.e., supply), at (b) the "optimal" price. In this thesis, we propose to drill down into the details of how media acquisitions can be performed using different metrics.

In traditional media contexts, i.e., print, radio and TV, supply has been acquired "directly", i.e., the media has been sold and purchased as directly traded merchandise. A newspaper, or a radio/TV station, for instance, would employ an ad sales team, who would go directly to the demand sources – ideally the brands who wanted to design campaigns for their products (e.g., Nike, for a new shoe), but in practice often to the ad agencies that run campaigns on behalf of brands – and sell ad inventory. The agencies, in turn, would employ media buyers, whose job consists of negotiating with media ad sales teams to acquire inventory at the lowest price. Buying media, traditionally, has been no different than buying, say, enterprise software.

The onset of digital advertising, on the web, brought about the emergence of a different way to acquire media – without human intervention, through an automated process, commonly referred to as Programmatic Media Buying (PMB) (Ebbert, 2012). PMB refers to the process of executing automated media buying (through platforms such as ad exchanges, agency trading desks, and Demand Side Platforms (DSPs) or Supply Side Platforms (SSPs) – much more on this later), rather than through traditional methods of manual advertising Request for proposal (RFP) and negotiation. PMB has been touted as the future of media planning and buying, especially in the online digital segment. A September 2012 Forrester report ("The Future of Digital Media Buying") (Joanna and Greene, 2012) asserts that media professionals not engaging in programmatic buying will be in jeopardy of losing their jobs through obsolescence. However, the adoption on PMB in real-life has not been encouraging. According to Walter Knapp, the executive VP of platform revenue and operations at Federated Media, one of the world's largest digital advertising networks, only about 10% of all display ads that are seen online have been traded programmatically (Vega, 2012). There are well understood reasons for this, relating to (a) resistance to change on the part of digital media buyers and (b) the predictability and durability of traditional digital media, i.e., web-sites (Datta, 2013).

In this thesis, we made a case that PMB will be the primary way of acquiring inventory in mobile apps using different metrics like popularity and audience. In other words, it can be asserted that while, in the case of traditional digital media (web sites), the promise of PMB has not yet been realized, in the case of mobile apps, PMBs will be the only way to go. Not only is this of critical importance to the practitioners in the massive and rapidly growing mobile advertising market, it has substantial impact on researchers as well. This is because, as we argue in this thesis, to enable PMB in mobile, a series of novel problems will need to be addressed. The identified novel problems would help advertisers in implementing PMB for advertisements. The focus of this thesis and potential contributions for Mobile Ad Eco System and Mobile App Eco system are described in the following section.

1.2. Research Focus and Potential Contributions

In this thesis we focused on identifying signals which would help in programmatic media buying in designing effective mobile ad campaigns. We propose that ad campaigns would be more effective when there is a way to determine the popularity signals in real time and when there is a way to pass the relevant audience information of mobile apps from which ad requests are coming. In the following sub sections we argued that how programmatic buying of social media popularity of an app, real time popularity rank computation of mobile apps and app audience information are vital in designing ad campaigns.

1.2.1. Programmatic media acquisition based on social media popularity

Social media platforms have emerged as a leading medium of conducting social commentary. Here users remark upon all kinds of entities, events and occurrences. As a result, organizations are starting to mine social media to unearth the knowledge encoded in such commentaries. Applications that can benefit from such knowledge discovery are many: trending topic discovery, sentiment analysis of consumer products and gauging public reaction to political campaigns are to name a few. A key requirement of a majority of such applications is the timely identification of social media mentions related to specific entities of interest, like products, persons or events. Such identification is well understood to be difficult due to a number of reasons, including (a) real-time discovery of relevant social media mentions given their massive rate of generation (Jansen et al. 2011; One Riot, 2009) (b) handling multi-lingual posts and (c) interpreting highly cryptic social media mentions, driven by brevity constraints (Dent and Paul, 2011)

Since twitter has emerged as the leading platform for social commentary (Kwak et al., 2010) we will explore this problem further using twitter, i.e., the real-time identification of microblog postings ("Twitter posts") that contain references to pre-specified entities of interest. For example, someone might wish to identify tweets that talk about the movie "Harry Potter and the Deathly Hallows: Part 2." A particularly interesting issue is to evaluate the popularity of specific mobile apps by analyzing the social conversation on them. Clearly, twitter posts related to apps are an important segment of this conversation and have been a main area of research in this context. In this thesis we propose a scientific approach which can be used to measure the popularity of apps in social media and ultimately this signal can be used by advertisers when designing ad campaigns. Therefore in this thesis, we propose that programmatic buying of social media popularity of mobile apps will enhance the effectiveness of mobile advertisements.

1.2.2. Programmatic media acquisition based on popularity ranks

Mostly in all traditional media segments (print, TV, Radio and Web) advertisements are delivered based on the popularity persistence which is a vital underpinning principle across all of these ad-ready media segments but will not hold for mobile apps since the popularity of apps is highly transient. Though mobile app popularity is highly transient, mobile app advertisers knowing popularity rank is very vital, as they are interested in placing their advertisements in apps with the greatest reach; clearly an app ranked high possesses greater reach than a lower ranked app. In a nutshell, the knowledge of how popular an app is with respect to its cohorts, commonly referred to as popularity ranks, can be greatly beneficial to app advertisers and for other constituents as well .

The crux of the motivation of the second study is that: accurate app popularity ranks (which will be simply referred to as "ranks" in the rest of this thesis) are not generally visible, and very difficult to compute. There are two primary reasons for this.

1. There is no single universally agreed upon metric that accurately measures app popularity. Rather, there exist multiple legitimate popularity signals attached to an app, which are dynamic and frequently conflicting. We discuss further on this in Chapter 3 of the thesis.

2. It turns out that the native app stores rank apps, both overall and by categories. These ranks, which will be referred to as *native store-category ranks* (NSCR), are easily visible to any interested party, either directly from the app stores, (who are highly secretive and zealously guard their NSCR computation methodology (Sarah, 2013) or via any of the popular app information aggregators such as App Annie (Bertrand, 2013) and Distimo (Hoogsteder, 2013). NSCR are important and in that they provide visibility to apps and greatly impact their future popularity. However, it is well known that the NSCR by themselves are open to manipulation and are commercially driven[13] and there is much evidence that it does not represent true popularity ranks, and are often misleading[14]. In other words, the NSCR, at best, represent a noisy popularity signal, but cannot be used as an accurate measure of popularity rank.

Without doubt, knowing app ranks is very useful, yet, as just described, these are notoriously hard to get to know. Thus motivated, in this thesis wes explore the problem of computing accurate app popularity ranks in real time as a second study. We propose that it would be vital for advertisers knowing the app

[13] http://www.macrumors.com/2012/02/06/apple-warns-developers-not-to-manipulate-app-store-rankings/ & http://venturebeat.com/2012/07/03/apples-crackdown-on-app-ranking-manipulation/

[14] The NSCR mechanism is not transparent. It is not clear whether the apps are ranked based on number of downloads, active installations and/or average rating? (Girardello and Michahelles, 2010). Moreover, NSCR are also criticized for being commercially driven – often an app launched the same day, which has had no time to build sustained popularity history, will show up in the top 5 of a rank list. Professional manipulators also abound, employing automated bots or hiring people to rate an app highly, a large number of times, which has been proven to boost their ranking. The, app stores themselves are battling with these ‘software bots’ or armies of human users who download apps in mass.

popularity ranks in real time and programmatic buying of this information would increase the effectiveness of mobile advertising.

1.2.3. Programmatic media acquisition based on audience profile

The problem of audience profiling in the context of web display ads is simple due to the presence and active use of web cookies to track the audience history. However, the absence of such cookie-driven capabilities in mobile phones makes it difficult to do the same for mobile app users. This creates an opportunity for a new breed of companies like AppAnnie, Mobilewalla and AppsFire, which collect volumetric data on mobile app audiences. For instance, an advertiser serving an ad at any given point of time must be supplied information regarding the trending apps for the preferred audience segment at that very moment, failing which, the advantages of real-time bidding are lost. Thus motivated, in this thesis we address the following research question "In absence of panel-based techniques to measure app popularity, how to design and develop newer strategies to capture unbiased audience data from mobile apps?" We in this thesis aim to resolve this challenge by proposing a non-panel based reliable scientific technique. We propose that buying audience profile of a given mobile app would increase the effectiveness of reaching the correct audience via mobile advertisements.

1.3. An Explanatory Framework for Ad-Serving in Mobile Apps

We in this section introduce the various frameworks for ad-buying and ad-serving that are being used presently and conclude with an in-depth description of the mobile app ad framework. We start off by reviewing the ad-serving framework for traditional media in the following section.

1.3.1. Direct Ad-Serving Framework (Traditional Media)

In traditional media like television, radio and the print, the process of ad serving occurs through a process of direct trading, as we have described previously. The publishers expose their available inventory to potential advertisers who in turn buy them to fulfill their ad budgets. The publishing house has a fixed number of slots which are up for sale and the advertisers are well aware of the value of these slots on account of their popularity persistence. This framework is illustrated in Figure 1.

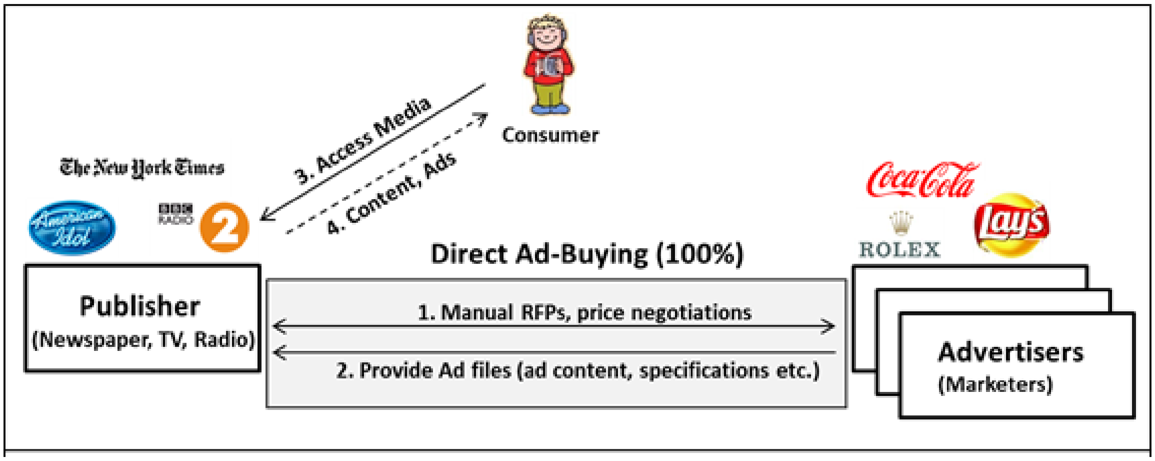

Figure 1 - Direct Ad Serving Framework

The key players in this framework include the publisher (a radio show or a television channel in this case), the media consumer and multiple advertisers willing to market their products. As with most direct trading strategies, the nittygritties of the advertising contract are discussed and agreed upon prior to the launch of the media. This offline process (Step 1 in the figure above) usually entails an exchange of advertising RFPs, negotiation of prices and finalization of the ad campaign duration. Once this is done, the advertiser generally provides the publisher with the ad specific information (Step 2), including the content of the ad that is to be displayed (or played out in case of a radio show).

I. Pitfalls with the Direct Ad-Serving Framework

With the advent of the web, the direct ad-serving framework started showing signs of inefficiency. The web differed significantly from the conventional media in one important aspect. Unlike TV, radio shows or newspaper columns, impressions on websites were not deterministically known. The amount of impressions served by the website was essentially contingent on the number of times the website was opened by the users. Furthermore, the barriers to entry into the online space were minimal. This inspired a sharp increase in the number of publishers and in turn, the amount of available inventory too. At the same time, the cost per impression nosedived. This led to a glut of cheap and unsold inventory on part of the publishers (called remnant inventory). The advertisers too were growing increasingly disillusioned with the quality of impressions being served as they received very little information about the target audience.

1.3.2. Online Ad-Serving Framework (World Wide Web)

With the advent of panel based audience measurement techniques for internet users, more and more information about consumer segments for each website started becoming available to the advertisers. As a result, advertisers could differentiate websites which had cemented their reputation as popular sites versus those that were not popular. Even though the number of websites grew rapidly, the popularity of the top few websites persisted. We have described this phenomenon in detail in the previous section as the popularity persistence of conventional media. Due to the persistent popularity of websites, advertisers could directly purchase their ad inventory using the direct ad-serving framework described above. A sizeable bulk (~ 90%) of the online-ads being displayed today are purchased through this technique while a very small section of advertisers (~10%) use alternate strategies to buy their ads. We illustrate below, the online ad-serving framework.

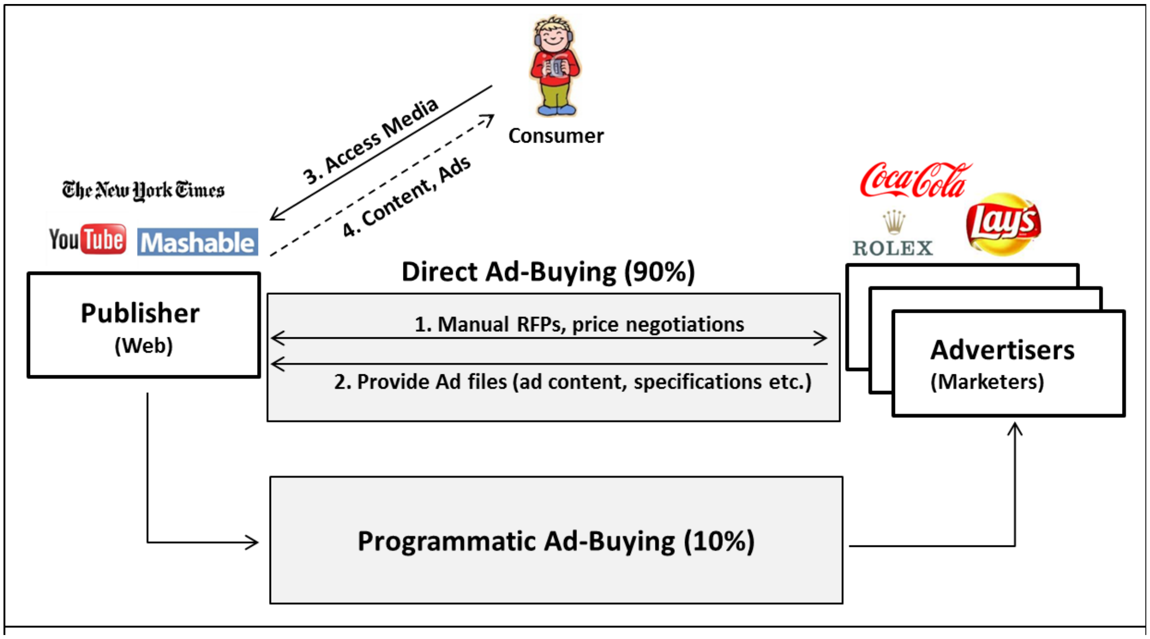

Figure 2 - Web Ad Serving Framework

As can be seen from the ad-serving framework in Figure 2 above, about 90% of the ads are purchased offline using exchange of RFPs and price negotiations. For example, consider the case of Rolex, a market leader in luxury watches willing to advertise on a popular video sharing site in the US using a video-ad creative. Through market research, Rolex obtains information from data aggregation companies which suggest that Youtube is the most popular video sharing site in the US. Rolex also observes that Youtube has successfully retained its rank within the top 3 video sharing sites for over 4 years now. This provides the confidence to negotiate an advertising contract with Google (which owns YouTube) much prior to the day its actual ads become visible. Such patterns of advertisement buying dominate the present online-advertising ecosystem. Once the bulk of the premium inventory is sold in a direct fashion, the publishers like Youtube might decide to sell-off any remaining inventory (low-quality impressions) using real-time bidding frameworks. A minority of advertisers purchase online ads in such a programmatic manner. We describe, the various steps of programmatically serving ads in detail while discussing the final framework of serving ads – that of mobile in-app advertisements.

1.3.3. Mobile Ad-Serving Framework (Mobile Apps)

The web ad-serving framework seems to work well for the web as the popularity of websites is fairly stable over a period of time. The same is not true for mobile apps as we have demonstrated in the previous section. Thus, in such an ecosystem, where app popularity cannot be predicted in advance, advertisers need to adopt an alternate media buying strategy. Advertisers would have to make their publisher selections in real time depending on the suitability of the impression as well as the current popularity of the app. We in this thesis suggest that the programmatic buying of ads, which has been fairly ignored for online-ads, is the only way out for the in-app advertisers. In Figure 3, it elucidates the framework that makes this possible. We observe that direct buying of ads is also a possibility for safe apps whose popularity remains fairly stable over a period of time. However, the number of such apps is extremely low in comparison to the total number of available apps in the major app stores. Thus, even though the minority of advertisers (~10%) might continue to use this mode of media buying, while the majority of advertisers (~90%) would have to adopt programmatic buying to remain profitable. Next, we introduce the framework for programmatic buying of mobile app ads.

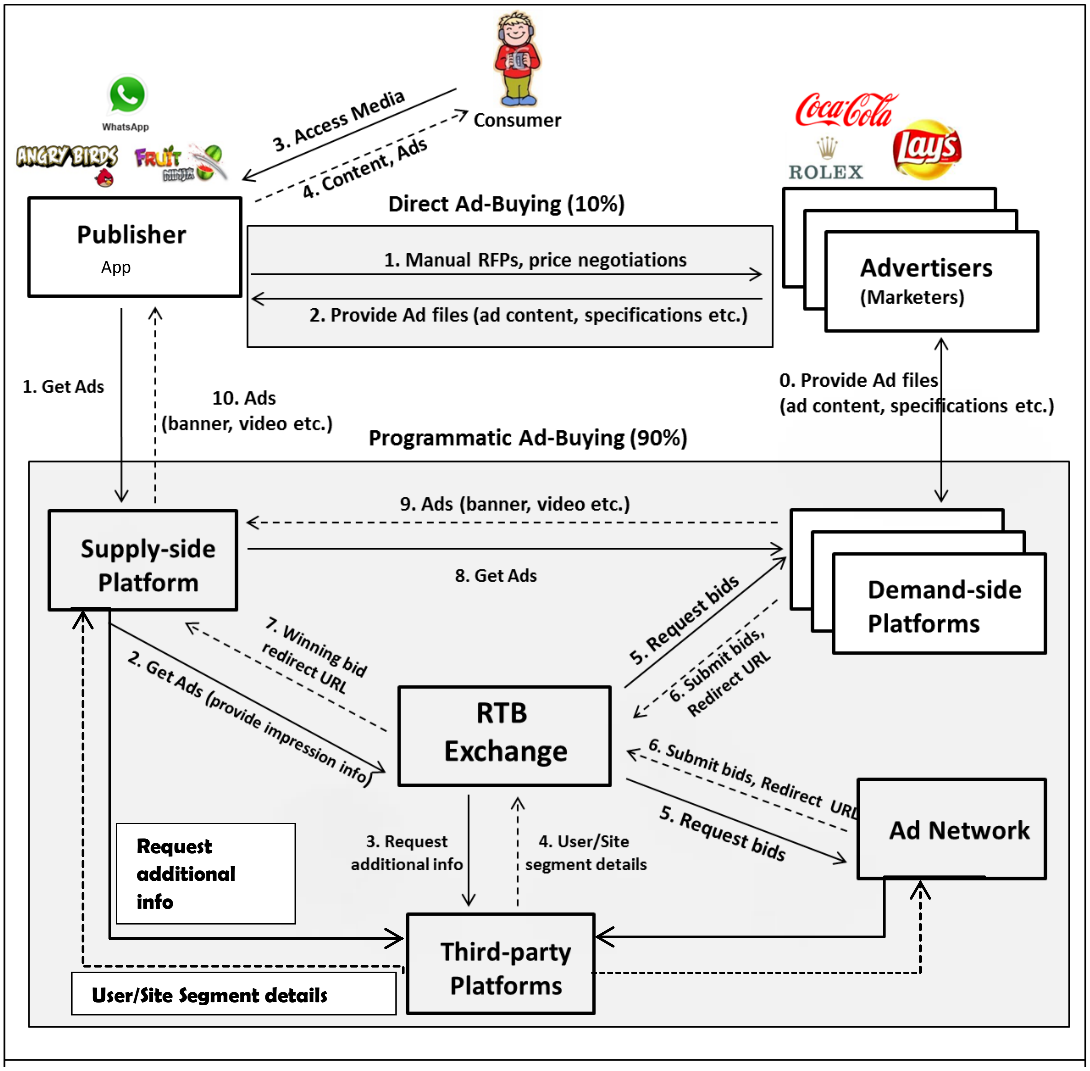
WhatsApp
ANGRY BIRDS
FRUIT NINJA
3. Access Media
4. Content, Ads
Consumer
Coca-Cola
ROLEX
Lay's
Publisher
App
Direct Ad-Buying (10%)
1. Manual RFPs, price negotiations
2. Provide Ad files (ad content, specifications etc.)
Advertisers
(Marketers)
1. Get Ads
10. Ads
(banner, video etc.)
0. Provide Ad files
(ad content, specifications etc.)
Programmatic Ad-Buying (90%)
9. Ads (banner, video etc.)
Supply-side Platform
8. Get Ads
Demand-side Platforms
7. Winning bid redirect URL
2. Get Ads (provide impression info)
5. Request bids
6. Submit bids, Redirect URL
RTB Exchange
6. Submit bids, Redirect URL
5. Request bids
Ad Network
Request additional info
3. Request additional info
4. User/Site segment details
Third-party Platforms
User/Site Segment details

Figure 3 - Mobile Ad Serving Framework

The programmatic buying infrastructure includes several new stakeholders - the ad networks, the supply and demand side platforms as well as the ad exchanges (Figure 3).

In the PMB scenario, an active app (running on a consumer's mobile device) makes an ad request call to a supply side platform (SSP) for mobile apps advertisements (Step1). SSPs have their Software Development Kits (SDKs) built into the app by the app developer, which makes an API call to the ad exchange to initiate the bidding process (Step 2). The API call generally includes information about the context (IP address, location, timestamp etc.), the device (the phone type, OS version, hardware IDs) and optionally the user information (gender, age etc.) that the app might have collected with consent from the user. The ad exchange, however, needs more details about the impression before it can pass it on to the members with a request for bids, but due to the absence of audience tracking cookies (e.g. SSP cookies, as in the case of web ads), the information deficit at the exchange becomes much more pronounced. This is where, third party data aggregation companies for mobile apps like AppAnnie, Mobilewalla and Appsfire provide the exchanges with additional data (demographics, prior usage patterns etc.) about the audience segments (Steps 3,4). The exchange now combines information obtained from the SSPs with this third party data and sends out bid requests to its member DSPs and ad networks (Step 5). The bid requests that are sent to the DSPs are made to conform to the Open RTB specifications charted by the Interactive Advertising Bureau (IAB) (IAB, 2011). The DSPs now analyze the impression that is up for bid and tries to match the various components of the impression viz. user demographics, locations information etc. to its own targeting requirements for each ad campaign. When it finds a campaign that shows a very high level of match (i.e. match on several components) with the impression characteristics, it returns a very high bid to the exchange, together with a redirect URL of its own ad server (Step 6). If, however, the DSPs find a poor match with the ad campaign requirements it might still place a bid, but with a much reduced value. The DSP bids are then compared at the exchange in real time and the marketing server URL from the winning bid is returned back to the SSP (Step 7). The winning DSP is then billed based on a second-price auctioning strategy (winner pays bidding price of second place bidder). The SSP now makes a call to the winning DSP's ad server to fetch the ad which is pushes to the appropriate publisher (Steps 8, 9, 10).

In addition to RTB exchange request for additional information from third parties, supply side platforms also can request for additional information before they send the ad request to RTB exchange. Same way demand side platforms and ad networks can also request for additional information such as user segments or user site details. Thus by enabling programmatic buying of mobile advertisements all the constituents in the mobile ad eco system can get benefited. Third parties can provide information like audience profiles for an app, social medial popularity of a given app and real time popularity of apps. Thus this thesis focuses on enabling programmatic buying by getting additional information such as social media popularity of mobile apps, real time popularity of mobile apps and audience profile of mobile apps from third party providers.

In the current state of the mobile ad ecosystem, both models of inventory buying viz. direct and programmatic coexist. However, we emphasize that in the mobile app ecosystem, direct trading of mobile ads always implies an inefficient use of inventory. We convince and point out that the time has come for the mobile advertising industry to embrace real time bidding on exchanges as the primary mode of buying and selling remnant as well as premium inventory.

We conclude this section with a brief discussion of the various pricing metrics that have evolved as a result of this real-time bidding and ad-serving framework. Traditionally marketers were generally billed based on CPIs and CPMs (Cost per Impression and Cost per Mile, where 'Mile' is Latin for the word thousand), wherein impressions were packaged in bundles of thousands or millions and billing was done at a bundle level. This was obviously inefficient when the marketer was interested in certain impressions in a bundle but not certain others. This spurred the advent of a newer set of metrics, namely, the CPC (Cost per Click), CPA (Cost per Action, also referred to as Cost per Sale and Cost per Lead) and CPI (Cost per Install, not to be confused with the earlier used Cost per Impression). An advertiser using an in-app banner ad would be more interested in knowing (and paying for) the number of customers who have clicked on its ad (the Click Through Rate) and perhaps even converted the click to a sale, than the number of casual visitors who have viewed the app. A related concept involves measuring number of customers who have viewed the impression, not clicked on it but later gone on to make a purchase from the marketer site. The number of such customers is measured by certain ad networks using a metric called the View Through Rate (VTR) and as is intuitively clear, this metric is one of the toughest metrics to evaluate in the advertising world.

The process of programmatic buying of media opens up several research questions spanning multiple areas of study. Wepropose that three different programmatic buying based solutions will enhance effectiveness of mobile app advertisement.

1.4. Thesis Organization

This chapter (Chapter 1) explains the motivation behind the thesis, an explanatory framework for ad serving in mobile apps, the specific research questions to be answered in the mobile app ad eco system, and the purpose of the thesis. The rest of the thesis is organized as follows:

Chapter 2 discusses the first study of this thesis. It outlines the importance of identifying social media mentions related to mobile apps and details the proposed solution. Further, evaluation of the proposed methodology is also discussed. Chapter 3 focuses on second study of this thesis which describes on the computing popularity ranks for mobile apps This proposed methodology is based on Deviation based OWA (DOWA), which is an adaptive and dynamic weighting scheme, and the weights attached to various features dynamically change based on importance of it in the ranking mechanism. Relevant literature pertaining to this study and the efficacy of proposed approach are also discussed in this chapter.

Chapter 4 presents the final study which describes on measuring audience of mobile apps. It further discusses as the how the proposed text mining based approach can be a novel solution for the problem discussed earlier.

Chapter 5 concludes the thesis with a discussion on the impact of the studies and their implications for theory and practice and provides possible directions for future research.

Chapter 2
Study I: Programmatic media acquisition based on social media popularity

2.1 Background and Motivation

The Twitter platform has emerged as a leading medium for conducting social commentary, where users remark upon all kinds of entities, events and occurrences. As a result, organizations are starting to mine twitter posts to unearth the knowledge encoded in such commentaries. Applications that can get a number of benefits from such knowledge discovery, including trending topic discovery, sentiment analysis of consumer products and gauging public reaction to political campaigns. A key requirement of a majority of such applications is the timely identification of twitter posts related to specific entities of interest, like products, persons or events. Such identification is difficult to well understand due to a number of reasons, including (a) real-time discovery of relevant twitter posts given their massive rate of generation (Jansen et al., 2011; One Riot, 2009), (b) handling multi-lingual posts and (c) interpreting highly cryptic tweets, driven by brevity constraints (Dent and Paul, 2011).

In this work, we have explored the problem of identifying microblog postings which contain references to pre-specified entities of interest, specifically to mobile application. It can help to measure overall social media popularity of an app. For instance, if someone wants to identify tweets that talk about the movie "Harry Potter and the Deathly Hallows: Part 2".

Two key problems that need to be addressed to perform such identification arise due to (a) the practice of aliasing entity names and (b) naming conflicts that arise between the entity of interest and other objects. Aliasing, driven by the need to conserve space, is the practice of using a subset of complete entity names (such as "Harry potter", for "Harry potter and the deathly hallows: Part 2") to refer to the entity. Clearly, if the identification system was unaware of such aliasing, it would perform poorly. The second problem, i.e., naming conflicts arises from semantic overloading of entity names, and is a common problem in the general search area. For instance, a film historian seeking information about the movie "ten commandments" (a phrase with wide connotations) will find that a simple search with just the movie title yields an enormous amount of information not related to the movie. However, adding contextual clues to the title (e.g., "ten commandments movie", "ten commandments de mille", "ten commandmentsheston") would yield high quality results (Cui et al., 2003; Google Inc, 2011; Sarkas et al., 2009). In most cases (such as in regular internet search), the user performing the search is aware of additional context clues (such as the fact Charlton Heston played the lead role in Ten Commandments) and can easily expand the search term.

In Twitter, the aliasing and entity name conflict problems assume special significance as the brevity of twitter posts precludes the usage of traditional context clues. When searching for any entity type, the searcher has to face this problem often in the domain of mobile applications, which we explain further.

An interesting feature about mobile apps is their virality - most successful apps (e.g., Angry Bird, Talking Tom, Flashlight etc.) gained popularity not by explicit outbound marketing, but rather, through viral word-of-mouth diffusion. Consequently, social media plays a significant role in the success of mobile apps.

Given this context, we have attempted to evaluate the popularity spread of mobile apps by analyzing the social conversation on them. Twitter posts related to apps are an important segment of this conversation. However, when we tried to extract twitter posts related to specific apps we discovered that it was a difficult task, due, to the aliasing and name conflict problems. For instance when searching for tweets discussing the popular iPhone app titled “Movies by Flixster with Rotten Tomatoes -Free”, we found that tweeters typically aliased this app simply as “Flixster”. Then attempted to simply search for tweets containing the term “Flixster”. However, even this has proved to be challenging as it was discovered that “Flixster” is overloaded – it could refer to both the app or the website (http://www.flixster.com/) – it was not easy to discard the tweets referring to the website and retain those referring to the app. We found that these issues to be common across many apps. Clearly, unless these issues are addressed meaningfully, it would be impossible to perform the core task, i.e., extracting tweets referring specifically to apps.

Now, we present a strategy to reliably extract twitter posts that are related to specific apps, overcoming the aliasing and name conflict issues discussed above. Once relevant twitter mentions are identified for a given app it can be subjected to sentiment analysis and influential behavior on others. While we motivated by mobile apps, the techniques are completely general and may be applied to any entity class.

In the next sub section (2.2), we reviewed related literature pertaining to this study. In sub section 2.3, we described the solution approach. In sub section 2.4 we experimentally demonstrated the efficacy of the techniques and in sub section 2.5 we discuss the potential contribution of this study and future directions of this study.

2.2 Related work

In this section, we discuss the relevant literature pertaining to twitter post filtering mechanisms followed by the impact of word of mouth on product success. Further, in section 2.2.2 we detail the need for identification of the social media mentions related to a product or service to measure its popularity.

2.2.1 Twitter post filtering mechanisms

At present, several text filters are available in the market. Commercial solutions such as Tweet filter (TweetFilter, 2012), Filter Tweets (Filtertweets, 2012) and

Social Mention API (SocialMention, 2012), can be used to filter the text, based on exact keyword match.

Tweet filter (TweetFilter, 2012) is a browser plugin that runs on top of “twitter.com”. Using Tweet filter, tweets can be filtered by matching usernames, keywords, phrases or source. Filter Tweets (Filtertweets, 2012) is a browser based script for filtering tweets by a specific topic and it works only with the new version of Twitter. One of the features in Filter Tweets is filtering tweets that contain a set of terms. Social Mention (SocialMention, 2012) is a social media search and analysis platform that aggregates user generated content from more than 100 social media web sites including: Twitter, Facebook, FriendFeed, YouTube, Digg, Google+ etc. It allows users to easily track and measure what people are saying about a person, company, product, or any topic across the web’s social media landscape in real-time. Social Mention provides an API to filter the user generated contents based on the given keywords from the popular social Medias mentioned above.

All of the above-mentioned commercial solutions have similar characteristics. First, all of them work based on exact keyword match, however as described in the Section 2.1, mobile apps are seldom referred to with the full name in the twitter posts, so it will be difficult, if not impossible, to find twitter posts related to mobile apps using any of the three. In other words, these solutions do not address the aliasing or name conflict problems. We will demonstrate experimentally in Section 2.4.

Some of the academic research relevant to this problem are discussed below. Inherently, at the end, the aim is to classify each twitter post as whether or not it is related to a mobile app or not. Thus, at a high level the problem resembles as a classification problem. In this respect the Bayesian classification technique is worth mentioning. The study titled “An Evaluation of Statistical Spam Filtering Techniques” (Zhang, Zhu, and Yao, 2004) evaluates five supervised learning methods such as “Naive Bayes”,“Maximum Entropy model”,“Memory based learning”, “Support vector machine”(SVM) and “Boosting” in the context of statistical spam filtering. They have studied the impact of different feature pruning methods and feature set sizes on each learner’s performance using cost-sensitive measures. We have observed that the significance of feature selection varies greatly from classifier to classifier. In particular, we found SVM, AdaBoost, and Maximum entropy model to be the top performers in this evaluation, sharing similar characteristics: not sensitive to feature selection strategy, easily scalable to very high feature dimension and good performances across different data sets. In contrast, Naive Bayes (Lewis, 1998; Nigam, 1999), a commonly used classifier in spam filtering, is found to be sensitive to feature selection methods on small feature sets, and fails to function well in scenarios where false positives are penalized heavily. Many previous studies (Androutsopoulos et al., 2000; Sahami, Dumais, Heckerman, and Horvitz, 1998; Schneider, 2003) have revealed the popularity of “Naive Bayes” (Lewis, 1998; Nigam, 1999) in anti-spam research and found that it outperforms keyword based filtering, even with very small training corpora.

The paper by Sriram et al. (2012) has proposed an intuitive approach to determine the class labels and set of features with a focus on user intentions on Twitter. Their work classifies incoming tweets into categories such as News (N), Events (E), Opinions (O), Deals (D), and Private Messages (PM) based on the author information and features within the tweets. Their work is based on sets of features which are selected using a greedy strategy.

Sriram et al.'s (2010) work experimentally shows that their classification out-performs the traditional "Bag-Of-Words" strategy. Unlike this research, the proposed approach does not rely on supervised learning, thus it does not have the overhead of feature selection and manual labeling. In addition, proposed approach can be used to classify a tweet as referring to any mobile app out of an arbitrarily sized set of apps, unlike Sriram et al., who need a predefined exact number of categories into which they perform the classification.

In addition to classification of short text messages, integrating messages with meta-information from other information sources such as Wikipedia and Word-Net (Banerjee, 2007; Hu et al., 2009) are also relevant. Sankaranarayanan et al (Sankaranarayanan et al., 2009) introduced "TweetStand" to classify tweets as news and non-news. Automatic text classification and hidden topic extraction (Banerjee, 2007; Sriram et al., 2010) approaches perform well, when there is meta-information or the context of the short text is extended with knowledge extracted using large collections. This does not apply in these case for mobile apps

Currently, there are about one million mobile apps in the market (Mobilewalla, 2012). To classify each twitter post as related to one or more of these apps, or not at all related to any of the mobile apps, will require equivalent number of classes, i.e., 1,000,000 classes in the classification approach. Such a large number of classes are impossible to handle using existing machine learning and classification techniques such as Support Vector Machine (SVM) (Chang and Lin, 2001) and Artificial Neural Networks (ANN) (Fausett, 1994). Therefore, instead of applying a classification approach, in this study, we address the problem at hand using corpus based data driven approach.

2.2.2 Word of Mouth and Product Popularity

Word-of-Mouth (WOM) is found to be a major springboard to drive download activity of mobile apps (Kats, 2012). According to a MTV Networks Survey, app discovery is driven almost exclusively by the recommendation culture, and 53% of survey respondents reported that personal recommendations though WOM are important in deciding which apps to download while 52% relied on user reviews (PRNewswire, 2011). WOM is critical for app marketing because apps require social proof to truly stand out (Cohen, 2013). User generated reviews at app stores and posts from at Social Networking Sites (SNS) can be viewed as two type of electronic WOM (eWOM), which has recently attracted a great deal of attention among practitioners (Trusov, Bucklin, and Pauwels, 2009; Z. Zhang, Li, and Chen, 2012). Understanding the social influence of eWOM is important because it presents important insights into how eWOM via social media affect the hyper competition of apps and how advertisers could incorporate social media as an integral part of advertising campaign. Thus, advertisers can target the apps which are more popular on social media to make their advertising campaigns effective.

In the quest to understand electronic Word of Mouth (eWOM), there has been an emerging interest in studying the effects of eWOM from Social Networking Sites (SNS) (Trusov et al., 2009). Word of mouth (WOM) is the process of conveying information from person to person and plays a major role in customer purchase decisions (Richins and Root-Shaffer, 1988). In commercial situations, WOM involves consumers sharing attitudes, opinions, or reactions about business, products or services with each other. The emergence of Internet-based media has facilitated the development of eWOM, which is accessible to multiple people via online channels (Hennig-Thurau et al., 2004). Prior studies have examined the effects of eWOM on consumer product sales (Chevalier and Mayzlin, 2003) , consumer decision making processes (De Bruyn and Lilien, 2008), and attitude towards brands and websites (Lee, Rodgers, and Kim, 2009). Furthermore, there has been an emerging interest in examining textual metrics (such as sentiment valence) of consumer reviews and their influence on sales (Zhang et al., 2012) .

SNS represents an ideal tool for eWOM, as consumers freely publish, consume and disseminate product-related information in their established social networks composed of friends, classmates and other acquaintances (Vollmer and Precourt, 2008). An understanding of eWOM in SNS and online product reviews can enhance the knowledge of the effects of eWOM and provide valuable insights into social media advertising strategy (Chu and Kim, 2011). Thus, an investigation of SNS (such as Twitter) as an online tool for eWOM is timely and needed. One paradigm for studying the constant connectivity of SNS in the commercial area is called the attention economy (Davenport and Beck, 2001), where brands constantly compete for the attention of potential customers. In this attention economy, SNS is a new form of communication in which consumers can describe things of interest and express attitudes that they are willing to share with others in posts. Given its distinct communication characteristics, SNS posts deserve serious attention as a form of eWOM.

Rui and Whinston, (2011) proposed the SNS-based business intelligence system that utilize real time information from Twitter with sentiment analysis techniques. Having described the need for measuring the social media mentions related to a product or service, in this study we propose an approach to identify the relevant twitter posts related to mobile apps.

In the next section, we first describe the intuition behind this approach and then explain the algorithm in detail.

2.3 Solution Approach

In this section we provide the intuition behind proposed approach and then delve into the details. A precise statement of first study related this thesis is as follows: given app “A”, find twitter posts that refer to “A”. Then identified twitter posts can be subjected to popularity measurement using the metrics like number of mentions, sentiments, number of re-tweets etc. (Bollen, Mao, and Pepe, 2011; Kwak et al., 2010; Mathioudakis and Koudas, 2010). In order to identify the relevant twitter posts, we propose two steps namely “Alias Identification” and “Conflict Resolution”.

1. First we discover what alias is commonly used by users to refer to app A as names are often abbreviated in the length-restricted twitter posts (140 characters). For instance, the popular iTunes app “Doodle Jump -BE WARNED: Insanely Addictive”, is commonly referred to in twitter posts as “Doodle Jump”. This step is called as “Alias Identification” step.
2. After alias identification, we need to resolve name conflicts, i.e. make sure that the twitter posts it is found refer to the app and not to other objects with the same name. One particularly ripe area for conflicts is between mobile apps and a regular web application. To see this, one has to consider the popular iPhone app titled “Movies by Flixster with Rotten Tomatoes - Free”. It turns out that this app is commonly referred to as “Flixster”. However, a twitter post containing the term “Flixster” might be referring to the app, or, to the highly popular sister website. This study is of course

interested in the popularity of the mobile app. Similar issues arise in the case of the Facebook app, or the Google Translate app. This phase is referred as “Conflict Resolution.”

2.3.1 Intuition behind the Alias Identification Phase

To identify the appropriate alias of an app with name A, sub phrases contained in A that are the most meaningful and unique are identified. Such *meaningfulness* and *uniqueness* (described below) is judged in the context of a Social Media Corpus (SMC) which has been constructed by lexical analysis of a vast amount of data gathered from Social Media Avenues such as Twitter, Facebook and the user comments awarded to apps in the native app stores.

Meaningfulness: Intuitively, meaningfulness refers to the semantic content of a phrase. For instance, in the context of the app title “Doodle Jump -BE WARNED: Insanely Addictive”, the reader can easily see that the sub phrase “Doodle Jump” is more meaningful than, say “Be Warned”, or “Insanely Addictive”. From an information theoretic perspective, meaningful n-grams (***n-gram*** is a contiguous sequence of *n* items from a given sequence of text or speech. For E.g. a given sentence like “Colorless green ideas sleep” contains 5 uni-grams, namely “colorless” “green”” ideas” “sleep” and “furiously”) will exhibit higher collocation frequencies relative to individual occurrence frequencies of the constituent 1-grams. This ratio is defined as $Affinity$ in this study.

$Affinity$ measures the likelihood of co-occurrence of the constituent words in a phrase. Intuitively, if certain words in a phrase occur often in the presence of one another, they have high $Affinity$. For example, let us consider the following sentence: “Attention deficit hyperactivity disorder is more common in boys than girls, and it affects 3-5 percent of children in the United States.” Noun phrase extractor will extract “attention deficit hyperactivity,” “attention deficit hyperactivity disorder,” and “deficit hyperactivity disorder” as some of the noun phrases in this sentence. Intuitively, based on the set of noun phrases extracted, the most meaningful phrase in this phrase set is the 4-gram phrase “Attention Deficit Hyperactivity Disorder,” as compared to the other phrases, e.g., “attention deficit hyperactivity” and “deficit hyperactivity disorder”.

Formally, we define $Affinity$ as follows:

$$Affinity(P) = \frac{f(P)}{min_{\forall w_i \in P(f(w_i))}} \times (1 - \max(P1, P2)),$$

where $f(P)$ is the frequency of phrase P in the SMC and $min(f(w_i))$ is the minimum frequency across the words in phrase P. The term $(1 - max(P\,1, P\,2))$ computes the relevance of the phrase with respect to its neighbourhood. Higher the value of $(1 - max(P\,1, P\,2))$ is less relevant the word phrase is with respect to its neighbourhood. If there is a pre-word for phrase ‘P’ and the pre-word is not a stop word, then it is combined with phrase ‘P’ to generate the new phrase called P_{pre}. If there is a post word for phrase ‘P’ and the post word is not a stop word then it is combined with phrase 'P' to generate the new phrase called P_{post}. If a pre-word or post-word is $null$ then $P1,\ P2 = 0$. Here $P1,\ P2$ are measured as follows, $P1 = \frac{f(P_{pre})}{1+f(P)}$ and $P2 = \frac{f(P_{post})}{1+f(P)}$. The higher the value of $\left(1 - max(P1, P2)\right)$ is, the less relevant the word phrase is with respect to its neighbourhood. For example for a given sentence like the following "National University of Singapore is leading University in Asia as well in the world" possible 2 and 3 gram noun phrases can be extracted out of the given sentence are "National University", "University of Singapore". In this case if we consider the 3-gram phrase "University of Singapore" as an example, for this given phrase pre-word is "National", post-word "is" but "is" is a stop word in the English dictionary[15]. Thus $P_{pre} = "National\ University\ of\ Singapore\ "$ and there will not be any P_{post} for this given 3-gram.

For the app name "Doodle Jump - BE WARNED: Insanely Addictive!" , Table 1 shows the frequencies and A*ffinity* measurement of word phrases, which formally identifies the word phrase "Doodle Jump" as more meaningful than others. Note that the table does not show all phrases whose affinities are measured for comparison. For a particular *n* (*n* = 1. . .*N*, where N is the number of words in the name of the application as the respective mobile app store), it is taken all *n*-grams from left to right beginning with the first word and stopping at the (N − n+ 1)th word.

[15] https://sites.google.com/site/kevinbouge/stopwords-lists

Table 1 - Affinity Measure

Phrase	Affinity
Doodle Jump	0.097
Be Warned	0.062
Insanely Addictive	0.003
Doodle Jump - BE	0.027
BE WARNED: Insanely	0.024

We have conducted One sample t-test to make sure difference between the population mean affinities to sample mean affinity are significant. Population mean affinity was computed using all the possible word phrases extracted for dynamically growing app names. Then for a given app (E.g. Doodle Jump) top 3 affined word phrases are extracted with their affinities values using SMC. Results show that differences are significant between these groups significant at the p <0.05 level).

Uniqueness: The meaningfulness property, while useful, is by itself not adequate to resolve the problem. In order to see this one has to consider the following: Let hypothetically assume (perhaps due to sampling biases while corpus creation) that the sub phrase “insanely addictive” is as (or more) meaningful than “Doodle Jump”. This system, using meaningfulness alone, would then judge “insanely addictive” as the best alias for the app “Doodle Jump -BE WARNED: Insanely Addictive” – a patently bad choice (as “insanely addictive” might be used in the context of many other apps). The uniqueness property (used in tandem with meaningfulness) prevents this misjudgment, by ensuring that the selected alias is used often in the correct context, but rarely in alternate contexts. Furthermore, *Affinity* does not apply to 1-grams (since there is only one word in the extracted token) and it cannot be directly compared to the uniqueness property. As such, this step will help to choose between the most meaningful n-gram phrase and all other 1-grams such that the end result is both highly meaningful and unique. Thus, to quantify uniqueness, a slight modification is made to the well-known IR notion of inverse document frequency (*idf*) (Spärck Jones, 1972) for a word or word phrase. The traditional *idf* is defined as:

$$idf(P) \ = \left(\ log_2 \frac{|D|}{df(P)} \right),$$

where |D| is the total number of documents in the corpus and $df(P)$ is the document frequency of phrase P, namely the number of documents that contain phrase P in corpus.

Modified expression is as follows:

$idf(P)_t \ = \left(\ log_2 \frac{1}{1+tcount(P)} \right)_t$ where $tcount$ is the frequency of P as recorded by Twitter in the target time interval T and it has been done away with |D| because for all phrases, the number of documents in the corpus (in this case, number of tweets in Twitter's database) within the target time interval T will be the same. Since the purpose is to looking for the highest $idf(P)$ it does not matter what |D| actually (|D| earlier in the orginl formula and ignored in the modified formula). It will be retrieved phrase level $tcount(P)$ directly from Twitter. For instance, the idf of the phrase "Doodle Jump" in the corpus is 18.28 but the idf values of

"Doodle" and "Jump" are 14.2 and 7.6 respectively. Therefore, "Doodle Jump" has more uniqueness and rarity than the individual terms "Doodle" and "Jump".

2.3.2 Intuition behind the Conflict Resolution Phase

The alias identification step ensures that the best alias is selected, but does not guarantee that this alias will not have conflicts with other object names, as illustrated in the "Flixster" example above. The purpose of this phase is to minimize the error. The core idea is as follows: Assume an alias, say S, is context-overloaded. The objective is to identify the overloaded aliases and then rerun the core tweet search by using a new search term that consists of the alias and a few contextual terms that disambiguate the search (e.g., "flixster + iPhone"). The additional context raises the probability that the retrieved tweet is talking about the mobile app domain.

2.3.3 Details of Alias Identification Phase

As discussed in section 2.3.1, in this step we discover the alias A of an app A, based on its meaningfulness and uniqueness values. This procedure is shown in Algorithm 1 from steps 1-6. Here, step 1 extracts all sub phrases from A (using a parser (Apache, 2012)), and computes affinities of each sub phrase in step 2. Subsequently, in step 3, we extract the most meaningful (highest affinity) phrase. This phrase is then subjected to a uniqueness test in step 4 by comparing its idf to the idfs of all 1-grams derived from A. Based on this test, the selected alias A is returned.

After alias identification, the tweets containing this alias are considered Legitimate, while disqualified posts are marked as irrelavant. The legitimate tweets are then subjected to the conflict resolution phase, which we describe below, to ensure that these refer to the app, and not to other objects with similar labels.

2.3.4 Details of Conflict Resolution Phase

In order to ensure that legitimate tweets refer to mobile apps and not to alternate objects, we design a classification mechanism where we first identify dual purpose aliases (e.g., Flixster, Facebook) and then incorporate additional context. More specifically, it runs the k-means clustering algorithm (MacQueen, 1967) on all the idf values of the aliases A with k = 2, i.e. two clusters (higher and lower idf clusters). The two initial mean points for each cluster are the lowest and the highest idf values across all aliases. This is shown in Algorithm 1 in step 7. The result of the k-means classification will be two sets of aliases, a high idf cluster and a low idf cluster.

We can explain this by the following example: After partitioning the top ranked Android apps based on the idf values of their aliases, it is found “paper toss”, “pocket god”, “words with friends”,“ebay mobile”, “pandora radio” and “espn scorecenter” belonged to the high-idf cluster, indicating they exist only in mobile app domain. Conversely, “flixster”, “google earth”, “skype” “facebook”, “kindle”, “bible”, “flashlight”, “netflix”, “backgrounds” and “translator” are aliases with low idf values, indicating these names are used both in mobile apps and in other domains, such as web applications.

Algorithm 01 - Algorithm for retrieving exact query phrase to use on the tweet database to ensure high relevance

1. Generate set of all word phrases C of length 2, 3 or 4 of the app name A. For example, for the app name “Doodle Jump -BE WARNED: Insanely Addictive!”, some of the collocates will be “Doodle Jump”, “Be Warned” and “Insanely Addictive”.
2. Compute $Affinity(C_i)$ for each word phrase $C_i \in \ C$ as derived in Step1. For example, $Affinity(\text{“}Doodle\ Jump\text{”})$ = 0.09, $Affinity(\text{“}Doodle\ Jump\ Be\text{”})$ = 0.00068 and $Affinity(\text{“}Be\ Warned\text{”})$ = 0.06.
3. Identify the word phrase C_i^{max} that has the highest value of $Affinity(C_i)$. In the example, the highest value is for Affinity(“Doodle Jump”) = 0.07, thus C_i^{max}= “Doodle Jump”.
4. Compute the idf for C_i^{max} and all one gram word of the name A. In the example, $idf(\text{“}Doodle\ Jump\text{”})$ = 18.28, $idf(\text{“}Doodle\text{”})$ = 14.2, $idf(\text{“}Jump\text{”})$ = 7.6, $idf(\text{“}Warned\text{”})$ = 7.79 and so on.
5. Identify the word phrase that has the highest idf as computed in step 4. In example, “Doodle Jump” has the highest idf.
6. Return the word phrase identified in Step 5 as the alternate app name A' of the app A (phrase with the highest idf).

7. After running steps 1-6 for all app names, k-means clustering algorithm is applied on the idf values of the word phrases returned in step 6 with a k value of 2 and the initial means to be the highest idf and lowest idf values in the corpus respectively. This will yield two clusters, one that is $high - idf$ and one that is $low - idf$.
8. For all word phrases that are part of the $low - idf$ cluster, append extra context keywords before querying the tweet database. For all words phrases that are part of the high-idf cluster, it can be used the word phrases "as is".

For the aliases with higher idf, in their associated twitter posts, as there is a very high probability that the post refers to the mobile app.

For the aliases with the low idf values, it is incorporated additional filtering mechanisms, by adding additional keywords like "app", "Android", "iPhone", "iPod", "Apple" and "iPad". Tweets containing any of these additional keywords are considered relevant (Legitimate), otherwise it is categorized as irrelavant.

2.4 Experimental Results

In this section, we demonstrate the efficacy of the approach proposed for this study, which will be referred to as TApp. The idea is to evaluate the quality of the legitimate tweets produced. If a tweet refers to the appropriate mobile app, the result is correct, otherwise, for that particular tweet, the procedure has failed. Specifically, it is required to test for both Type 1 and Type 2 errors, i.e., how well it retains and how well it avoids the rejection of good tweets. First, comparison is carried out with Naïve Bayesian approach. Next, it is compared with one of the commercial platforms, Socialmention (SocialMention, 2012).

2.4.1 Comparison with Bayesian Approach

For a baseline comparison, the Naïve Bayes classifier (Lewis, 1998; Nigam, 1999) , a popular method for document classification in anti-spam research, (Androutsopoulos et al., 2000; Sahami et al., 1998; Schneider, 2003) has been used. Since the training input is pre-processed app names, token-based naive Bayes classifier is used to compute the joint token count in app description and category probabilities by factoring the joint into the marginal probability of a category times the conditional probability of the tokens given the category defined as follows.

It is widely used in text categorization task (Nigam, 1999) and often serves as baseline method for comparison with other approaches (Zhang et al., 2004). In the implementation of Naïve Bayes (using the Laplacian prior to smooth the Bayesian estimation, as suggested in Nigam, 1999 - In Laplacian smoothing we see every outcome once more than the acutal count) classification, a set of keywords are extracted from every twitter post and used those as the feature set. Based on the keyword occurrences in the twitter posts in the training data, probabilities are calculated. These probability values are used to classify the twitter posts.

Both the TApp and the Bayesian classification technique have been implemented using Java 1.6.

We have carried out all the experiments using a Windows 7 machine with quad core processor of 2.33 GHz.

In order to compare TApp with the Bayesian classifier, A set of “apps of interest” has been first selected – for this experiment, The top 50 “hot” android apps has been chosen using a popular mobile app search engine platform (http://www.appbrain.com/apps/ hot/). To create the test bed for these 50 apps, set of (~2000) tweets have been randomly selected from database of 14 million tweets and manually verified whether they contained references to these apps ($legitimate\ tweets$) or not ($spam$). In this fashion 1000 posts manually identified from the randomly selected tweets, consisting of 500 posts that refer to one of these 50 apps ($legitimate\ posts$) and 500 tweets that refer to mobile apps or internet web sites, but not any of the selected 50 mobile apps. Both the Bayesian classifier and the TApp approach have been applied on this test bed to classify these 1000 posts into Legitimate and Spam. In Figures 4 and 5, illustrates the histogram distributions of accuracy of the two approaches -Bayesian and TApp. As it can be seen from Figure 4, the Bayesian classifier identifies 337 out of the 500 Legitimate posts (a recall rate of 67%), whereas the TApp approach demonstrates a recall of 97.2% by correctly classifying 486 of the 500 Legitimate posts. Similarly, as portrayed in Figure 5, the Bayesian classifier wrongly identi-fied 174 of the 500 Spam posts as Legitimate, whereas TApp misidentifies only 23 of 500. Table 2 is presented with classical IR metrics such as precision, recall, true negative, accuracy and F-measure in both the cases. In all cases TApp significantly outperforms the Bayesian classifier (TApp scores above 90% in every case).

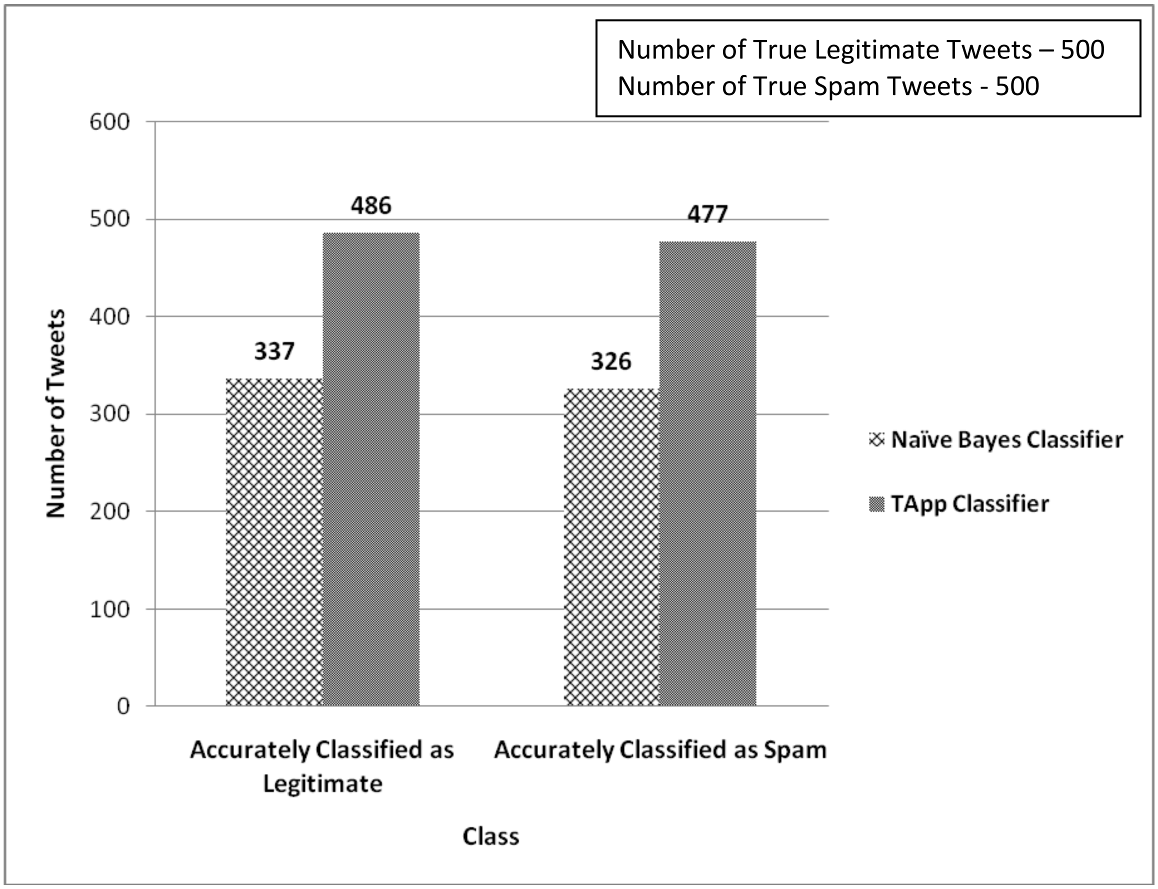

Figure 4 - Comparison of Accurate Classification

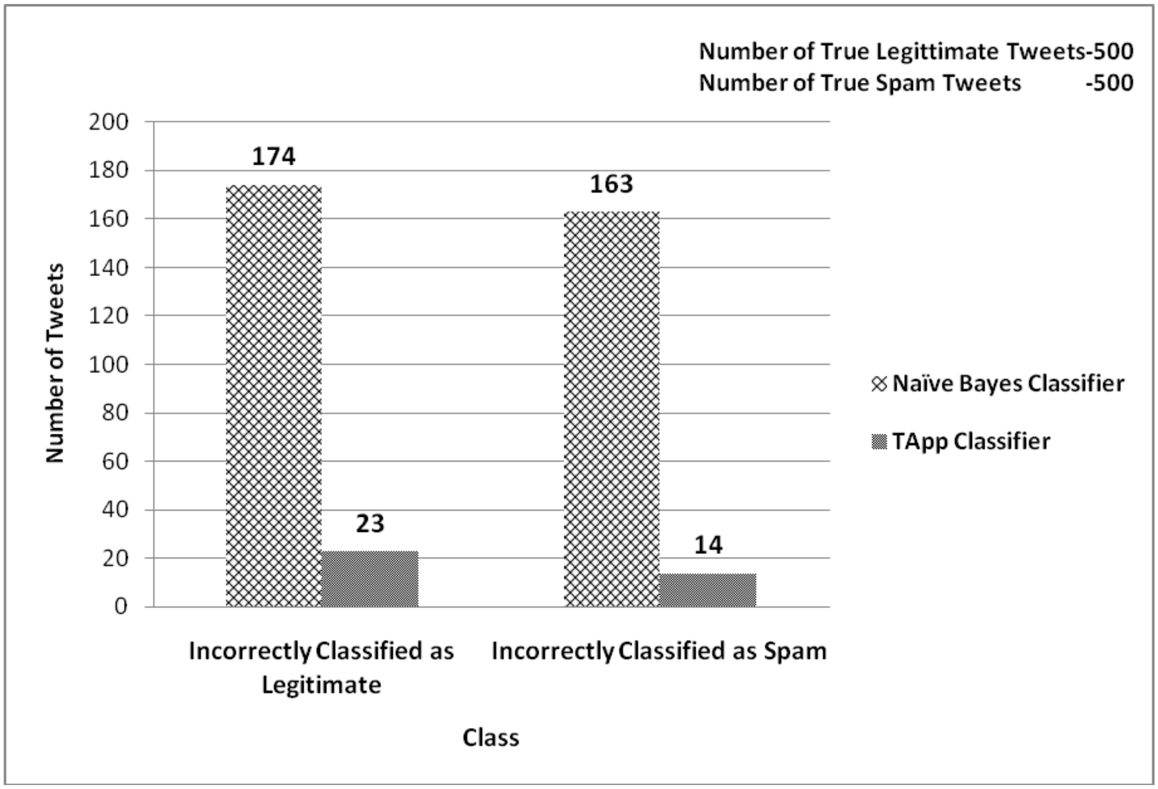

Figure 5 - Comparison of Incorrect Classification

Table 2 - Comparison of IR metrics in Bayesian classifier vs. TApp

Matrix	Naive Bayes classifier	TApp classifier
Precision	100 ∗ 337/(511) = 66%	100 ∗ 486/(509) = 95.6%
Recall	100 ∗ 337/(500) = 67%	100 ∗ 486/(500) = 97.2%
True Negative Rate	100 ∗ 326/(500) = 65.2%	100 ∗ 477/(500) = 95.4%
Accuracy	100 ∗ 663/(1000) = 66.3%	100 ∗ (963)/(1000) = 96.3%
F-measure	(2 ∗ 65.9 ∗ 67.4)/(66 + 67) = 66.7%	2 ∗ 95.6 ∗ 97.2/(95.6+ 97.2) = 96.4%

Table 3 - Comparison of Valid Tweets in "Socialmention" vs. TApp

	Store App Name	Alias Name	Using SM Valid	Using SM In-Valid	Using TApp Valid	Using TApp In-Valid
No Aliasing & name conflict	Cardio Trainer	Cardio Trainer	45	5	50	0
	Endomondo Sports Tracker	Endomondo Sports Tracker	20	8	27	0
	Google Sky Map	Google Sky Map	3	2	0	0
	Handcent SMS	Handcent SMS	38	11	16	0
	Instant Heart Rate	Instant Heart Rate	37	13	45	5
	Live Holdem Poker Pro	Live Holdem Poker Pro	47	3	50	0
	Lookout Mobile Security	Lookout Mobile Security	43	7	48	2
	Stardunk	Stardunk	39	11	27	8
	Total		313	69	312	8
	Accuracy		81.93%		97.5%	
Aliasing required, but no name conflict	Calorie Counter by FatSecret	Calorie Counter	3	4	50	0
	Documents To Go 3.0 Main App	Documents To Go	5	7	11	1
	Funny Facts Free 8000+	Funny Facts	1	1	48	2
	Bubble Blast 2	Bubble Blast	32	10	39	0
	Kid Mode: Play + Learn	Kid Mode	4	24	40	10
	Kids Connect the Dots Lite	Kids Connect The Dots	1	0	27	0
	PicSay - Photo Editor	Picsay	4	0	50	0
	Mango (manga reader) Free	Mango manga reader	10	3	43	6
	Pandora internet radio	Pandora	5	1	17	5
	SpeechSynthesis Data Installer	SpeechSynthesis	2	22	4	0
	Talking Tom Cat Free	Talking Tom Cat	28	14	48	2
	Vaulty Free Hides Pictures	Vaulty	1	0	26	0
	Waze:	Waze	2	1	50	0

	Community GPS navigation					
	Total		98	87	453	26
	Accuracy		52.97%		94.57%	
Both aliasing and name conflict required	Adao File Manager	Adao File Manager	1	0	1	0
	Advanced Task Killer	Advanced Task Killer	38	12	31	3
	Angry Birds	Angry Birds	30	20	46	4
	Backgrounds	Backgrounds	3	47	38	0
	Barcode Scanner	Barcode Scanner	6	44	48	2
	Bible	Bible	0	50	13	5
	Craigslist	Craigslist	0	50	2	2
	Drag Racing	Drag Racing	11	39	49	1
	Epocrates	Epocrates	33	17	50	0
	ES Task Manager	ES Task Manager	2	5	8	0
	ESPN ScoreCenter	ESPN ScoreCenter	34	16	43	6
	Facebook for Android	Facebook for Android	18	32	42	0
	FxCamera	FxCamera	18	5	17	0
	Google Maps	Google Maps	2	48	26	6
	Horoscope	Horoscope	3	47	15	0
	KakaoTalk	KakaoTalk	9	41	48	2
	LauncherPro	LauncherPro	31	19	50	0
	Mobile Banking	Mobile Banking	6	44	47	3
	Mouse Trap	Mouse Trap	2	48	9	0
	My Tracks	My Tracks	0	49	4	0
	NFL Mobile	NFL Mobile	19	31	50	0
	Ringdroid	Ringdroid	26	17	18	0
	Tap Fish	Tap Fish	22	28	47	3
	The Weather Channel	Weather Channel	3	47	45	5
	Tiny Flashlight + LED	Tiny Flashlight	24	26	47	3
	Total		358	845	819	41
	Accuracy		29.75%		95.23%	
Total			769	1001	1584	75
Accuracy			**43.44%**		**95.45%**	

2.4.2 Comparison with "SocialMention"

SocialMention(SM) (SocialMention, 2012) is the leading social media search engine. To demonstrate the effectiveness of the approach, we have decided to compare the accuracy the obtained results with those acquired from Socialmention. As discussed in the Section 2.2.1**Error! Reference source not found.**, the exact algorithm of Socialmention implementation is unknown. However, by observing different search results we have concluded that Socialmention uses an exact keyword matching approach to identify the twitter posts that contains the given keywords. In this experiment, the same set of 47 apps have been used in the previous experiment in Section 2.4.1. For each app, the tweeter posts related to that app in the previous one month were retrieved using both Socialmention API and the TApp approach. The objective of the approach is to automate the Twitter post retrieval for large number of mobile apps. So, the input to both Socialmention and the TApp approach is app names as found in native app stores. The Socialmention uses these original app names to find the twitter posts. TApp approach applies name aliasing and name conflict resolution to retrieve the relevant tweets. However, the app names are chosen to be such that 22 out of 47 require either no aliasing and/or no name conflict resolution. This was done to assess the effectiveness of the TApp technique in individually performing those 2 tasks.

In order to constrain the experimental data size, for each of the approaches if the number of posts for an app is more than 50, it has been considered only the most recent 50 posts. Next, passed on to the posts identified by both Socialmention and TApp along with the app names to two professional lexicographers. Each of the lexicographers has more than 5 years of experience of internet search optimization. They both worked together to arrive at an unanimous decision of which of these posts are “Valid” (i.e. the post is related to the respective app) and which of these are “invalid” (i.e. the post is not related to the respective app). The result is presented in Table 3.

As can be seen from Table 3, for many apps, the Socialmention platform has retrieved tweets that are not related to that app. In total only 43.44% of the total tweets retrieved by Socialmention has been identified as “Valid” post by lexicographers. Whereas, for TApp approach, the absolute number of invalid posts for each app is much smaller compared to the Socialmention. Overall 95.45% of the twitter posts retrieved by TApp has been identified as “Valid” by lexicographers. The total number of valid tweets retrieved by TApp is 1584 compared to 769 by Socialmention. So both in terms of accuracy and the coverage of retrieval, TApp has significantly outperformed the Socialmention.

Additionally, we observe that Socialmention works well in cases when there no aliasing of the app names and when there is no naming conflicts between the entity of interest and other objects. In these cases, Socialmention achieved 82.93% accuracy. For example, the extracted tweets for the apps “Live Holdem Poker Pro”,“Google Sky Map”,“Handcent SMS” and “Lookout Mobile Security” in both Socialmention and TApp are highly relevant because these names are only used in mobile app domain and there is no aliasing by users. One should observe that, even in these simple cases, where there is no name conflict and aliasing, the accuracy in TApp case is higher than that of Socialmention. The exact approach followed in Socialmention is unknown, so it is not sure of the reason behind this improvement; however it is anticipated that this is due to the generic keyword matching algorithms followed in Socialmention, vs. the phrase search using tweeter API followed in TApp.

In the second scenario, when the app names required aliasing, but no name conflict resolution, the Socialmention's accuracy in retrieving relevant tweeter posts was 52.97% compared to 94.57% in TApp approach. For example, the tweets extracted for the apps "SpeechSynthesis Data Installer", "Kid Mode: Play + Learn" and "Vaulty Free Hides" are mostly irrelevant or unfound because of aliasing practice of users when they post their tweets. These apps are typically referred to as "SpeechSynthesis","Kid Mode" and "Vaulty" in most of the tweets.

In order to show the effectiveness of TApp's entity name conflict handling, it is focused on the third category of app names, where both aliasing and name conflict resolution are required. When look at the valid tweet count for the apps "Drag Racing","Mouse Trap","Mobile Banking" and "My Tracks" in case of Socialmention, they are very low compared to the invalid tweet count. These app names are used outside the mobile application domain as well and so required name conflict resolution in TApp approach, which is clearly not done in Social-mention. For these type of app names, Socialmention had a pretty low accuracy of just 29.75% in retrieving relevant tweets compared to 95.23% accuracy in TApp case. This demonstrates the importance and effectiveness of both the aliasing and name conflict resolution steps in TApp.

2.5 Discussion and Conclusion

In this sub section we discuss the broader implications of the TApp approach. This study research falls in the design science research paradigm of Information Systems (Hevner et al., 2004). An artifact has been developed and which can successfully resolve name conflicts of app names in twitter posts. Effectiveness of the artifact has been demonstrated through experimental study and a comparison with a manual method. The two step approach out performs the benchmark Naïve Bayes classifier and a commercial implementation ("Socialmention" (SocialMention, 2012)) both on true negative and false positive errors.

In identifying social media mentions related to products in general and mobile apps in particular has important implications for advertisers, ad tech networks as well as for product owners. Once the social media mentions are identified, it can be further mined to analysis its content to predict the product's popularity. Being able to predict the social media popularity of items have tremendous value not only to service providers but also to marketers who would bid for ad-space on items with high potential popularity in order to maximize the exposure. Based on the proposed approach, programmatic buying of social media popularity can be enabled and effectiveness of mobile ad campaigns can also be improved.

TApp approach can be used to identify user generated contents across social media, which in turn can be later used to measure product's popularity. This approach can be utilized in many ICT research domains such as, identifying twitter posts related to brand monitoring in e-commerce, identifying the public opinions of e-participation, e-services and general e-government implementations by using the social media mentions, identifying students opinions of e-learning systems and analyzing the public views on digitizing the medical records of patients (Electronic Medical Records: EMR). Thus this approach is generalizable and broadly applicable across wide range of ICT research domains in general.

Further, we have addressed the problem of reliably identifying tweets related to mobile apps. In the process we further address the aliasing and name conflict problems inherent in the task. Proposed approach has been compared with Naïve Baysian approach and a commercial implementation ("Socialmention"). Proposed approach outperformed in all measures of accuracy compared to Bayesian approach and the "Socialmention". While the proposed approach has been validated in mobile app domain, the techniques are generally applicable to other domains as well.

Chapter 3
Study II: Programmatic media acquisition based on popularity ranks

3.1. Introduction

As discussed in section 1.2.2, in all traditional media segments (print, TV, Radio and Web) advertisements are delivered based on the persistence of popularity which is a vital underpinning principle across all of these ad-ready media segments but will not hold for mobile apps since the popularity of apps is highly transient. Though mobile app popularity is highly transient, for mobile app advertisers knowing popularity rank is very vital, as they are interested in placing their advertisements in apps with the greatest reach; clearly an app ranked high possesses greater reach than a lower ranked app. In a nutshell, the knowledge of how popular an app is with respect to its cohorts, commonly referred to as popularity ranks, can be greatly beneficial to app advertisers and for other constituents as well.

While the app market is undoubtedly massive, it is dominated by a very small number of the most popular apps. It turns out that 10% of apps command 90% of revenues (Crofford, 2011) and 80% of all app downloads (App Brain, 2012). For publishers and app developers, being at the top of the popularity heap is the Holy Grail, leading to both fame (downloads and engagement) and fortune (revenue).

More specifically, if apps were to be ordered on the basis of popularity, the most coveted objective for publishers is to be positioned as high on that list as possible (say within the top 100 of its category, or more preferably the top 10). It turns out that such popularity ranks are greatly useful for all participants of the app ecosystem, namely the **advertisers, consumers** and **publishers/developers**.

For advertisers[16], knowing popularity rank is vital, as they are interested in placing their advertisements in apps with the greatest reach; clearly an app ranked high possesses greater reach than a lower ranked app.

For app consumers, adrift in the impossible-to-navigate media ocean that native app stores have become, thus knowledge of popularity ranks can help app discovery (Ryan, 2011). For developers and publishers, popularity ranks can be an important method of self-assessment (i.e., how am I doing?) as well as a means to perform competitive analysis (i.e., how am I doing with respect to my competitors?). But perhaps the biggest impact of an accurate knowledge of popularity ranks is the enablement of effective app monetization. To see this, consider, in turn, the examples of the publishers of a paid app and a free app. For the paid app publisher, popularity impacts the number of downloads, which in turn determines revenue. For the publisher of the free app, the major monetization opportunity arises from enabling the delivery of in-app advertising campaigns – the higher the popularity of an app, the larger its *audience reach*, and

[16] The mobile advertisement market is $3B in 2012, but doubling every year (Opera 2012)

consequently the greater the per impression revenue[17] (and overall dollars) it can command. In summary, the knowledge of how popular an app is with respect to its cohorts, commonly referred to as popularity ranks, can be greatly beneficial to virtually every entity involved in the app domain.

Now we proceed to crux of the motivation for this work: accurate app popularity ranks (which shall simply be referred to as "ranks" in the rest of the chapter) are not generally visible, and very difficult to compute.

There are two primary reasons for this.

1. There is no single universally agreed upon metric that accurately measures app popularity. Rather, there exist multiple legitimate popularity signals attached to an app, which are dynamic and frequently conflicting. An extended discussion will be made of this very soon in this chapter.
2. It turns out that the native app stores rank apps, both overall and by categories. These ranks, which it will be referred to as *native store-category ranks* (NSCR), are easily visible to any interested party, either directly from the app stores, (who are highly secretive and zealously guard their NSCR computation methodology (Sarah, 2013) or via any of the popular app information aggregators such as App Annie (Bertrand, 2013) and Distimo (Hoogsteder, 2013). NSCR are important in that they provide visibility to apps and greatly impact their future popularity. However, it is well known that the NSCR by themselves are open to manipulation and

[17] Typically this is measured in *Cost Per Million* (CPM) dollars

are commercially driven[18] and there exists much evidence that they do not represent true popularity ranks, and are often misleading[19]. In other words, the NSCR, at best, represent a noisy popularity signal, but cannot be used as an accurate measure of popularity rank.

Without doubt, knowing app ranks is very useful, yet, as just described, these are notoriously hard to get to know. Thus motivated, the problem of computing accurate app popularity ranks is explored in this chapter.

3.1.1 Problem Details

In order to make the problem more concrete, and to make clear some of the notions (e.g., NSCR) which have been alluded to at a high level, we delve into a little more detail and provide with an example.

We have mention above that the core problem in computing app ranks is that the visible metrics that are generally used to estimate popularity are often conflicting and dynamically changing. Let's now dig a bit deeper into this. Consider two commonly used features of app popularity, native store-category rank (NSCR) and average number of daily social mentions (ADSM). NSCR represents the rank awarded by the native store in which the app is listed, e.g., if app A has a NSCR

[18] http://www.macrumors.com/2012/02/06/apple-warns-developers-not-to-manipulate-app-store-rankings/ & http://venturebeat.com/2012/07/03/apples-crackdown-on-app-ranking-manipulation/

[19] The NSCR mechanism is not transparent. It is not clear whether the apps are ranked based on number of downloads, active installations and/or average rating? (Girardello and Michahelles, 2010). Moreover, NSCR are also criticized for being commercially driven – often an app launched the same day, which has had no time to build sustained popularity history, will show up in the top 5 of a rank list. Professional manipulators also abound, employing automated bots or hiring people to rate an app highly, a large number of times, which has been proven to boost their ranking. The, app stores themselves are battling with these 'software bots' or armies of human users who download apps in mass.

of 3 in the ‘reference category’ in the iTunes app store at a time *t* in Denmark, that means that if a Danish user viewed a list of reference apps (e.g., Google Search, Free Translator, Bing) in iTunes at time *t* on his/her apple device, he/she would find A in the 3rd ordinal position in that list. Similarly, a high number of social media mentions is generally regarded as a good indicator of the popularity of consumer products (Asur and Huberman, 2010; Chevalier and Mayzlin, 2006; Duan, Gu, and Whinston, 2008), specifically for apps in this case. The average number of daily social mentions (ADSM) is a measure of this (Oghina et al., 2012). A method seeking to rank apps based on a weighted average of their NSCR and ADSM values would find that these features often move in opposite directions. For instance, on 29th of Oct 2012, the app “Pandora Radio”, listed in the overall leader board category in the iTunes United States app store, experienced a NSCR drop from 26 to 32, but enjoyed an ADSM rise from 10 to 91 (Mobilewalla, 2012). How does one compute an “overall” popularity measure in this case? To complicate matters, these metrics are temporally highly dynamic – in the iTunes app store for instance, the NSCR are continuously changing (much like equity prices in a stock exchange) and the rate of social media mentions (say on Twitter) could be continuously changing as well. To effectively use a weighted-average ranking method to compute overall ranks, one needs both a way of determining relative weights, and account for relative intensities of change. For the “Pandora” example quoted above for instance, we need to know which to prioritize over the other, NSCR or ADSM, a difficult decision to make. Even under the unrealistic assumption that enables to come up with a way to statically determine such a priority, it is still needed a way to account for the fact that NSCR only dropped 6 ranks (relatively small change) while ADSM increased by 81 (relatively large change). The following point are made: when ranks are computed based on multiple features, which are (a) dynamic, (b) conflicting and (c) whose relative contributions to the overall rank value are impossible to determine, coming up with a meaningful weighting scheme is a tough task. However, a mechanism which can discover the "true" ranks of apps would be extremely useful. Such a mechanism would need to quickly adapt to the dynamic and conflicting changes that occurs around apps and commercially unbiased.

Given this context, in this Chapter, we propose the design of a rank discovery mechanism for mobile apps, at the heart of which is an adaptive and dynamic weighting scheme, *where the weights attached to various popularity signals are themselves dynamic*, i.e., they change to adapt to the changes in the underlying signals. While the invented method is used to generate popularity ranks of mobile apps in this work, it can be employed to rank order items in any scenario where the rank determination signals are dynamic and the relative importance of these signals is unknown.

This approach is based on a well-known class of multi-criteria decision making (MCDM) technique called the Ordered Weighted Average (OWA) technique (Yager, 1988). In many MCDM scenarios the final "success" scores of the various alternatives are obtained through an aggregation process which computes the weighted average of the various decision factors, known as features. Often, in the real world, the relative weights of these features are not known and evaluating them is a difficult task. OWA techniques provide the best known methods to compute feature weights, when they are not known a-priori. Since the problem fits this case, the core OWA philosophy would appear to apply well.

Unfortunately, a direct application of this technique does not work, as OWA methods yield static weight vectors, i.e., once weights of various features are determined they do not change. This will not work in this case due to the continuously changing nature of the features; it will be demonstrated shortly through a detailed example. In response, a major new OWA variant, called DOWA (Deviation-based OWA) is designed where the weighting scheme itself is dynamic. DOWA represents a substantial extension of OWA and, we believe, this study introduces an important new class of MCDM solutions.

DOWA is evaluated in two ways. First it is demonstrated its “absolute” effectiveness, by comparing how its output compares with the “ground truth”. In this test, it is observed that DOWA is typically within 10% of ground truth values, demonstrating excellent accuracy. Second, DOWA’s accuracy has been evaluated against the premier current OWA variant known as PFLQ (Proportional Fuzzy Linguistic Quantifier) (Yager, 1988). These experiments demonstrate remarkable results across a variety of metrics (absolute deviation, root-mean-square-error, 90th percentile etc). DOWA grossly outperforms OWA, beating the latter by at least a factor of 2:1. By almost any measure DOWA is shown to perform exceedingly well, and represents the first provably good solution to the important app rank estimation problem in particular, and any dynamic feature-driven ranking problem in general.

The rest of the chapter is organized as follows. In the section immediately following, we discuss the relevant literature pertaining to this work. After that, we describe the proposed approach following the intuition discussion. In the App popularity model section, we identify feature variables for ranking mobile apps. Then, we present the experimental results demonstrating the quality of the approach and Thereafter we conclude the chapter.

3.2 Related Work

This section will cover the technology and scientific trends which we have briefly discussed in the Introduction section earlier, but not in greater detail. Ranking mobile apps is a special case of ranking consumer products possessing dynamic feature spaces, such as movies, books and TV shows. To understand the state-of-the-art we have performed a careful review on two broad research themes: (a) research on popularity based ranking of consumer products, and (b) the different methods used in the "core" ranking computation.

3.2.1 Ranking Consumer Products

A number of methods have been developed and introduced to measure the ordinal ranking of products based on popularity. Li, Bhowmick, and Sun, (2010) developed an approach to predict ordinal ranking of products, reviewed online product review websites such as *Epinions*[20] and *Blippr*[21]. First, the authors created an ordinal rank based on number of reviews received, positing that reviews are considered a strong indicator of popularity (Amblee and Bui, 2007) – this yielded an initial simple popularity ranking scheme. On top of this, the authors layer on additional signals like rank history, average ratings and variation of the ratings received from various users to forecast future ranks, using a time series based forecasting model. The forecasted rank is a weighted average of various product features, determined statically.

Mohanty and Passi, (2006) have proposed another method to rank products based on (a) online ratings, (b) customers' own disclosed preferences of product features, and (c) the ordinal rank of the product in search engine query results. Fuzzy logic has been applied to quantify customers' linguistic preferences. Customers' preferences and products features derived from available online product ratings are summed using **Ordered Weighted Averaging** (OWA) (Carlsson and Fuller 1997) to derive the overall product rating. This overall product rating is then combined with the search engine rank to compute the rank of the product. As in the work by Li, Bhowmick, and Sun, (2010), the weights generated are static, and therefore unable to fulfill the requirements.

Yin et al. (2012) recommended a *Conformer-Maverick (CM)* model to rank *potentially popular items*. Here the authors propose that if a product has received

[20] http://www.epinions.com/?sb=1

[21] http://mashable.com/category/blippr/

positive votes from the *conformer* group and negative votes from the *maverick* group, it should be highly popular. Based on this observation, the authors developed two ranking mechanisms - Aggregation-based ranking and Q-based ranking. The former predicts the vote for each user and aggregates them to predict the overall rank of a product. In contrast, Q-based ranking directly estimates the item's popularity degree and the corresponding rank. This approach predicts the future rank of products. In a way, votes can be regarded as ratings and it would appear that this method might be applicable to the mobile app scenario. However, mobile apps are complex multi-feature objects where ratings alone are not enough to indicate popularity. Moreover, only a small number of apps receive user ratings, complicating further the potential to apply this approach.

Ghose and Ipeirotis (2007) established an approach which ranks product reviews, rather than the product itself. Their objective is to quantify how effective product reviews are. They proposed and analyzed two ranking mechanisms for product reviews - (i) consumer-oriented ranking mechanism which ranks the product reviews according to their expected helpfulness to consumers and (ii) a manufacturer-oriented ranking mechanism which ranks the product review according to their impact on expected sales volume rank of the product. Their approach, relying quite heavily on qualitative analysis, creates a statistical model and estimates the model based on data collected from Amazon. Like virtually all other work in this area, this is a static model.

Having described a number of significant existing works on ordinal ranking of the products based on popularity, we reviewed relevant complementary literature that does not perform strict ordinal ranking, but, nevertheless, suggest strategies for discriminating among a set of items.

3.2.2 Item Discrimination Methods

Product ranking is complicated by conflicting features that contribute to the rank. Feng, Hwang, and Dai (2009) suggested the Rainbow Ranking System to solve this problem in e-commerce. The idea of Rainbow ranking is not to perform ordinal ranking of products, but, rather, creating a number of bands of products where products in a higher band are superior or equal to the products in lower band across all the features. Each band may contain several products and the approach does not differentiate across products in a single band. This approach is suitable to narrow down consumer choices (i.e., solving the product discovery problem) across multitudes of online products in an E-Commerce environment.

By studying the trend of votes for items in "Digg.com" and "YouTube", Szabo and Huberman (2010) analyzed the "evolution" of item popularity and predicted the future popularity growth of an item. In essence, they present a method for predicting the long-term popularity of online content based on early measurements of user access. The approach does not compute any ordinal or relative ranking of items. Each item is treated individually to predict its future popularity growth, as measured by a number of views of the item.

To predict popular items, a content-based technique has been recommended in Yu, Chen, and Kwok (2011). In this work, a textual item is split into meaningful words which in turn, form a feature vector of the item. Then, a classification machine is trained to predict, given an item's feature vector, the likelihood that this item will be popular.

Product reviews represent an important determinant of product popularity. In this context, there exists a stream of research in marketing literature that analyzes the positive relationship between online product reviews (aka word-of-mouth) and product sales. This research clearly demonstrates an association between positively rated products (such as a book) on a website and subsequent sales of the product on that site (Chevalier and Mayzlin, 2006). While this work does not detail a popularity ranking scheme for products, it ends up identifying features that have positive impact on product popularity, which forms an important input in this work.

Based on the above, we can conclude that the existing product ranking approaches use either product features or product reviews. A majority of existing literature are based on aggregating individual features by pre-determined static weights based on regression on a training data set or summing up with pre-determined static weights via approaches like the OWA approach (Yager, 1988, 1993). This approach is a significant departure from the extant mechanisms in that we recognize the inadequacy of using fixed weights in product ranking, and propose an entirely new “dynamic weight” variant of the OWA approach. Finally, in this section, we review the OWA approach, which forms the starting point of this study methodology.

3.2.3 OWA Approach

In multi-criteria decision problems, different decision criteria are typically weighted differently. For instance, the purchase decision for a vehicle might depend on its price, gas mileage, and passenger capacity. However, for young professionals who have just secured their first jobs price might be the most important criteria, while for a family, it might be passenger capacity. One of the most difficult issues in multi-criteria decision problems is to determine what weights to ascribe to the different factors. To solve this issue, Yager (Yager, 1988) introduced an aggregation technique based on the notion of ordered weighted average (OWA), widely regarded as the state-of-the-art method for multi-criteria decision making.

Intuitively, OWA works as follows: it accepts as input the values corresponding to a set of decision factors (referred to as *features*) and orders them (typically according to their strength). Then, it assigns an weight to each feature, based on its ordinal position in this ordered list – the feature in position 1 gets the highest weight, the feature in position 2 the next highest and so on. Finally, based on the assigned weights and the feature values, an aggregation operation is performed, yielding a specific success value corresponding to a given input vector. As an example, let us consider the problem of MBA student admission in a business school. Let's assume the candidates are applicants indexed A, B, C ... and the decision criteria are: (a) GMAT score, (b) Reference letters, (c) Undergraduate GPA and (d) work experience. For each candidate the feature values would be coded into a vector and the vectors would be input into the OWA method. The OWA method would then compute a weighting scheme for the 4 features by examining the values in the input vectors and creating an ordering as described above. Finally, for each student (represented as a vector of feature values) it would perform an aggregation and output a "success" score based on which admission decisions would be made.

Formally, a n-dimensional OWA operator is a mapping

$F: R^n \rightarrow R$ that has an associated n-dimensional weight vector W ,where $W = (w_1, w_2, \cdots, w_n)^T$ such that

1. $w_i \epsilon [0,1], \quad 1 \leq i \leq n$

2. $\sum_{i=1}^{n} w_i = 1$

Furthermore,

$$F\ (a_1, a_2, \cdots, a_n) = w_1 b_1 + \cdots + w_n b_n \ \ (Equation\ 1)$$

Ordering the arguments work in the way as shown in Equation 1 where b_j is the j^{th} largest element of the bag $R^n = < a_1 \cdots, a_n >$ and $b_1 \geq b_2 \geq \ldots. \geq b_n$. For example let assume, $W = [0.4, 0.3, 0.2, 0.1]^T$ then, $F\ (0.7, 1, .0.3, 0.6) = (0.4)(1) + (0.3)(0.7) + (0.2)(0.6) + (0.1)(0.3) = 0.76$

The actual aggregation performed by an OWA operator depends upon two factors - ordering of the feature arguments and the determination of the weight vector. Ordering of the arguments works in the way as shown in Equation 1 where b_j is the j-th largest element of the bag $(a_1; \dots\dots; \ a_n)$ and $(b_1; \dots\dots; \ b_n)$. The key here is to comprehend that the weights of OWA *are not associated with any particular value* a_j *; rather they are associated with the ordinal position of* b_j.

The most critical task in the OWA technique is to determine the weight vector. It turns out that several approaches have been employed to compute the exact values of weights, such as, maximizing entropy (O'Hagan, 1988), minimizing maximum disparity (Amin and Emrouznejad, 2006) and minimizing variance (Fuller and Majlender, 2003). However, one of the leading method to compute W is technique proposed by Yager and Filev (1994) as follows.

$$W_j = \left\{ \begin{matrix} \frac{1}{n}\ (1-\alpha) + \ \alpha\ , j = \ 1; \\ \frac{1}{n}(1-\alpha)\ , j = \ 2, \dots., n \end{matrix} \ , \alpha \ \in [0,1] \right\} \ (Equation\ 2)$$

Therefore,

$$F\ (a_1, a_2, \cdots, a_n) = \ \alpha\ Max(a_j) + \frac{1}{n}\ (1-\alpha) \sum_{j=1}^{n} a_j \ \ (Equation\ 3)$$

One characteristic of the OWA technique is that given a particular decision scenario, it produces a static weight vector – for instance, for the MBA student admission example it would assign a specific fixed weight to each of the 4 factors. As discussed previously, this does not work for context described above, i.e., computing app ranks. Therefore in this chapter, we propose a new OWA operator specifically developed for the scenario.

We describe some general properties and measurements related to OWA operators in the following subsection. Subsequently, we will use these to demonstrate the efficacy of the approach.

Properties of the OWA Operator

A known property of the OWA operator is that it can be mapped to the MAX, MIN or arithmetic MEAN operators based on the form of the weight vector W. Specifically, when $W = [1, 0, 0, \cdots, 0]$, the OWA operator will reduce to a max operator. When $W = [0, 0, 0, \cdots, 1]$, the OWA operator will reduce to a min operator. When $W = [1/n, 1/n, \cdots, 1/n]$, the OWA operator will be an arithmetic mean operator.

In Yager (1993), author has introduced the concept of dependent and independent OWA operators.

DEFINITION 1:

Independent and Dependent OWA operator: *An OWA operator is called* ***independent*** *if derived weights are associated with particular ordered positions of the aggregated arguments, and have no connection with the values of the arguments. An OWA operator is called* ***dependent*** *if the weights are determined based on the values of the input arguments.*

Thus for an independent OWA operator, $w_j = f_j(b_1, b_2 \cdots, b_n)$

$$OWA(a_1, a_2 \ \cdots, a_n) = \sum_{j=1}^{n} f_j(b_1, b_2 \ \cdots, b_n)\, b_j (Equation\ 4)$$

Yager(1988) showed that both independent and dependent OWA operators hold the *commutativity* and *idempotency* property, and are bounded by the Max and Min operators. However, independent OWA operators have the *monotonicity* property, while dependent OWA operators do not. Later in this chapter, wewill demonstrate that the proposed OWA operator is a dependent OWA operator. Further, we will show that the proposed OWA operator is bounded by the *commutativity* and *idempotency* properties like any OWA operator.

Measures of OWA - orness and dispersion

To analyze the relative importance accorded by an OWA operator to each of its input feature values, Yager (1988) introduced two measurements - *orness* and *dispersion*. For a given weight vector *W*, *orness*(*W*) characterizes the degree to which the OWA aggregation is like an 'or' operation and it is measured as,

$$orness(W) = \frac{1}{(n-1)} \sum_{i=1}^{n} (n-i) * w_i \ \ (Equation\ 5)$$

Clearly, $orness(W)$ is a real value between 0 and 1. When $orness(W) \ = 0$, the aggregated value yielded by the associated OWA operator reduces to the minimum feature value, signifying that the only "important" feature is this minimum feature. Conversely when $orness(W)$ = 1, the aggregated value is the maximum feature value. Generally, $orness(W) >$ 0.5 indicates the OWA operator awards more weight towards the higher feature values, while $orness(W) < 0.5$ indicates the OWA operator accords more weightage to the lower feature values. Needless to say, $orness(W) = 0.5$ indicates the OWA operator gives uniform importance to all features.

The second important measure, referred to as the dispersion (or entropy) of the aggregation, is defined as the degree to which W takes into account all the information encoded in the arguments during the aggregation. It is defined as,

$$dispersion(W) = -\sum_{i=1}^{n} w_i \ln w_i \ (Equation\ 6)$$

where $0 \leq dispersion(W\ ) \leq$ ln(n). Since dispersion provides a degree to which the information in the arguments is used, when *orness* = 0 or 1, the *dispersion* is “zero”. When w_i= 1/n (a uniform distribution), the dispersion is maximum, i.e., $ln(n)$. The concept of dispersion is similar to Shannon’s entropy (Shannon, 1948). The more disperse the W the more of the information about the individual criteria is being used in the aggregation of the aggregate value.

In the later part of the chapter, we will compare the *orness* and *dispersion* measurements of the approach to those of the PFLQ operator.

Having described the basics of OWA approach, now we describe the extended OWA technique in the next section.

3.3 Proposed Solution

We have designed an approach, called DOWA (deviation based OWA), to compute popularity based ranks for mobile apps. DOWA belongs to the OWA family of multi-criteria decision making techniques (described earlier in OWA approach section) and is based on a new OWA operator that has been created, called DOWA operator. This represents a major departure from extant OWA operators in that its associated weight vector W_{dowa} is dynamic, unlike the static, or fixed, weight vectors yielded by current techniques. We will describe the proposed DOWA approach and the DOWA operator, first by providing the underlying intuition and then delving into details.

3.3.1 Intuition

At the outset it is important to understand why current "fixed-weight" OWA schemes will not work for computing mobile app popularity ranks (or, indeed, the popularity ranks of any product possessing similar characteristics).

As discussed before, OWA techniques aggregate a variety of features to compute an overall *success metric* by determining the relative importance of these features in contributing to the overall "success". Another way of looking at it is to model the success metric as a *composite signal*, composed by the aggregation of a number of individual feature signals. In this model, the OWA operator yields a relative weighting of the base signals, which in turn yields the value of the success metric.

Now we consider how this plays out in this case. Here the success metric is the assignment of a *rank* value to each member of a collection of apps indicating their relative popularities.

The signals are specific features of apps that contribute towards their popularity. Let's assume, simplistically, that the three key popularity determining signals/features are:

- *The rank of an app in its native app store (i.e.NSCR)*: Clearly, a lower rank value (say 5 in the lifestyle category in the US iTunes store) would be associated with higher popularity that a higher rank value (say 10 in the lifestyle category in the US iTunes store).
- *Ratings of an app in the app store*: An app with a higher rating (say 4) would appear to be better liked, i.e., more popular, than one with a lower rating (say 3.5), everything else being equal.
- *Number of Reviews in the app store*: Intuitively, an app with a greater number of reviews would appear to have garnered more engagement, and therefore be more popular than another app with fewer reviews, everything else being equal.

Note that the values of these features are continuously changing – when assume the granularity of observation is a day, then, for an app, it's Rank (i.e.,NSCR), Ratings and Number of Reviews change daily. Indeed for the top (popular) apps the values could, and do, change dramatically.

In traditional OWA techniques, the weight of each of these features would need to be predetermined based on the strength of the individual signals. Thus, if the normalized value of store ranks were higher, on average, than the normalized value of the number of reviews, the store rank would be awarded a higher weight than the number of reviews. This works well for traditional application scenarios such as MBA admission case (discussed earlier).

What makes it unsuitable for ranking apps is the fact that the relative signal strengths themselves are continuously changing. Imagine, for instance, a situation where, based on existing data, a traditional OWA operator has determined that store rank is the strongest popularity signal, awarding it the greatest weight. Subsequently for an app, it may happen that its rank remains relatively unchanged for a period of time, but it starts getting an inordinately large number of reviews. Intuitively, it is clear that this signifies an increase in popularity for this app (clearly, many more people are engaging with the app than before). However, a static weighting scheme might fail to capture this sudden increase by greatly over-weighting the store rank feature (which still might be stronger on a normalized basis) and, simultaneously, under-weighting the review count feature. In other words, while a rapid increase in review count is clearly a stronger popularity signal at this time than an unchanged store rank, traditional fixed-weight methods might fail to capture this.

This problem is equivalent to the "near-far problem" problem in wireless communication systems. The "near-far problem" is a condition in which a receiver captures a strong signal and thereby makes it impossible for the receiver to detect a weaker signal (Rappaport, 2001), whereas the weaker signal may encode more important messages than the stronger signal.

The above discussion is abstract – we provide a real example below.

Consider the iconic game app titled “Doodle Jump” (DJ) that has regaled youngsters and veterans alike since the early days of smartphone apps. In Table 4, we provide the numerical values for the three popularity signals for DJ for 14 days in March 2102. In particular, we provide *normalized values* for store rank (*StoreRank*) in the US iTunes store in the Games category, rating score (*AVRScore*) and number of reviews (*AVRCount*) for DJ for the period March 1 through March 14 of 2012. We will now examine the effect of applying a traditional OWA technique on this data.

In applying OWA in this case, let us assume that the “success metric” of an app is computed by aggregating the values of signals. In other words, the success metric for DJ for March 6th, 2012 would be computed based on the feature values of the app on that day itself.

Aggregation can be done based on the OWA operator discussed earlier (Equations 2, 3).

$$SuccessMetric\ (\text{“}DoodleJump\text{"}\ ) \\ = w_1 \times (AVRCount) + w_2 \times (AVRScore) + w_3 \\ \times (StoreRank)$$

Here the weight vector $w = [w_1, w_2, w_3]$ is computed based on the technique discussed in Equation 2.

The success metric values, i.e., the computed OWA score of “Doodle Jump” from 1st March to 14th March is given in Table 4 in the *O*WA Score column. It is very evident that the score is dominated by the *AVRScore* feature, because that has the highest normalized value across all 10 days.

We note, however, that there has been a significant change in the strength of the *AVRCount* signal during this time period. This is evident in two ways: (a) its value has increased from 0.63 on 1st March to 0.92 on 14th March, almost a 50% increase, and (b) there is a consistent upward trend in the values. In contrast, *AVRScore* demonstrates remarkable consistency across the entire time period, indicating the signal strength remains constant. The paradox here is as follows: even after such significant enhancement of the *AVR-Count* signal, the absolute value of *AVRScore* remains higher throughout. As a result, when ordering the features based on its normalized values *AVRScore* will be placed first by traditional OWA, after which the *StoreRank* and *AVRCount* will be ordered. Therefore, *AVRScore* will receive the highest weight and will dominate the overall OWA score throughout this range. Due to this, in spite of the strong upward trend of *AVRCount* during the specified interval, the OWA score for DJ had a very nominal change from 0.84 to 0.92 at that interval, basically indicating that popularity remained practically unchanged. However, intuitively, we cansee that there was a significant real change in popularity, as one of the three signals went up substantially, while the other two remained steady. Thus the OWA approach of ordering features according to its normalized values fails to capture real changes in signal strength, and, therefore, provides inadequate quantification of the overall success metric (popularity, in this case).

Based on the above discussion, the intent is to create an OWA weighting mechanism that is sensitive not only to the absolute values of signal strength, but also to changes in relative strength. We will provide the details of the approach in the next section, suffice it to say here that preferential treatment will be accorded to features with changing signal strengths. Specifically, an additional (or lesser) weight will be awarded to a feature demonstrating consistent upward (or downward) trends, compared to a feature whose signal strength remains more or less unchanged. We will recognize the features with dynamic trends, as opposed to features whose signal strengths remains static by computing the standard deviation across historical value of a feature. Subsequently, the features will be ordered based on this standard deviation value, rather than the normalized value of the feature. Adopting this approach, the *AVRCount* receives the highest weight, followed by *StoreRank* and finally *AVRScore* for the given period. In Table 4, the scores computed by the proposed DOWA method is presented. As can be seen the

Table 4 - Time Series Data for Doodle Jump

				OWA Weight (According to Rank positions)				DOWA Weight			
Date	**AVRCount**	**AVRScore**	**Store Rank**	**1**	**2**	**3**	**OWA Score**	**AVRCount**	**AVRScore**	**StoreRank**	**DOWA Score**
2012-03-01	0.63	0.92	0.74	0.83	0.08	0.08	0.88				
2012-03-02	0.62	0.92	0.74	0.83	0.08	0.08	0.88				
2012-03-03	0.64	0.92	0.75	0.83	0.08	0.08	0.88				
2012-03-04	0.63	0.92	0.75	0.83	0.08	0.08	0.88				
2012-03-05	0.62	0.92	0.76	0.83	0.08	0.08	0.88	0.3333	0.3333	0.3333	0.767
2012-03-06	0.63	0.92	0.74	0.83	0.08	0.08	0.88	0.3333	0.3333	0.3333	0.763
2012-03-07	0.65	0.92	0.75	0.83	0.08	0.08	0.88	0.5000	0.2940	0.2060	0.75
2012-03-08	0.71	0.92	0.75	0.83	0.08	0.08	0.89	0.5000	0.2530	0.2470	0.773
2012-03-09	0.75	0.92	0.76	0.83	0.08	0.08	0.89	0.4903	0.2500	0.2597	0.795
2012-03-10	0.8	0.92	0.78	0.83	0.08	0.08	0.90	0.4409	0.2500	0.3091	0.824
2012-03-11	0.82	0.92	0.79	0.83	0.08	0.08	0.90	0.3686	0.2500	0.3814	0.834
2012-03-12	0.85	0.92	0.81	0.83	0.08	0.08	0.91	0.1291	0.3709	0.5000	0.856
2012-03-13	0.87	0.92	0.82	0.83	0.08	0.08	0.91	0.0664	0.4336	0.5000	0.867
2012-03-14	0.92	0.93	0.83	0.83	0.08	0.08	0.92	0.2270	0.5000	0.2730	0.9

DOWA score (0.78 on 6th March to 0.95 on 10th March) presented in Table 4 clearly represents a better quantification of Doodle Jump's popularity.

Having demonstrated why existing OWA approaches might fail in the context and also described the intuition of the proposed approach, we provide the details of the DOWA approach below.

3.3.2 Details of DOWA

In this section, we describe the approach; titled *Deviation based OWA (DOWA)* in detail, based on the intuition described in Intuition Section. The first step is to define the new OWA operator that will serve as the basis of the DOWA technique.

DEFINITION 2:

DOWA Operator: *The DOWA operator of dimension n is a mapping* D: R^n = R, *that has an associated* n *dimensional vector* $w = (w_1, w_2, \cdots, w_n)^T$ *, such that.*

$w_i \epsilon [0, 1]$, $0 \leq w_i \leq 1$

$\sum_i^n w_i = 1$

The DOWA operator denoted by D, works as follows:

$$D((u_{1,}a_1), (u_{2,}a_2), \dots \dots, (u_{n,}a_n))_t = \sum_{j=1}^{n} w_j \, \tilde{a}_\sigma{}^{(t,k)}{}_{(j)} \; (Equation\ 7)$$

where each argument of D, $(u_{i,}a_i)$; $1 \leq i \leq n$, is called a DOWA tuple, *t* denotes the current time, and $\sigma_{(j)}$ denotes a permutation of $(1,2,..,n)$

At a high level, reader's attention is drawn to the difference between the classical OWA operator F, defined in Equation 3, and the DOWA operator D defined above. The inputs to F are simply the features or signal values, denoted as a_i. The arguments to D, are not only the feature values, but also an associated *order-inducing* variable u_i, that indicates the relative change in signal strength across all input features (details below). While the classical F operator would yield the same weighting scheme given the same feature input values, the DOWA operator, in contrast, is sensitive to the change in signal strengths and might yield different weighting schemes even when presented with the same a_i values at different times. We describe the details below.

As mentioned above,$\left(u_{i,}a_i\right)$; $1 \leq i \leq n$, is a DOWA tuple. Within each such pair $u_{i,}$ is an order inducing variable based on the standard deviation values of the input features, and a_i is the argument value corresponding to the i^{th} feature. Further, t is the current time and $(\sigma_{(1)}, \sigma_{(2)}, \sigma_{(3),} \dots, \sigma_{(n)})$ is a permutation of $(1, 2, \cdots, n)$ such that $u_{j-1} \geq u_j$ for all $j = 2, \cdots, n$. Here $u_{i,}$denotes the standard deviation value of argument a_i in the time interval *(t-k) to (t-1)*. Moreover, ã$_{(j)}$is the average value of argument $a_{(j)}$in the time interval *(t-k)* to *(t-1).* In DOWA, the parameter k is an adjustable window size in days. In this approach ã$_{(j)}$ argument will be ordered based on its standard deviation values. It is interesting to identify both positive, (upward) and negative (downward) trend. Therefore, simply using the standard deviation itself to identify the trending behavior of a feature will not work, because standard deviation values could be higher when there is a sudden drop or a rise in feature values. Therefore, to identify the direction of the trending behavior in a feature, we employ the *average* value in addition to the *standard deviation*, i.e., we check whether the average value of a feature from time (t-k) to (t-1) is less than or equal to average value of the same feature from time (t-(k/2))to (t-1). This is achieved by imposing the constraint $\frac{2}{k}\sum_{j=1}^{k/2} a_{\sigma j} \geq \frac{1}{k}\sum_{j=1}^{k} a_{\sigma j}$ in Definition 2. This allows us to order features based on the magnitude of relative upwards trends.

Having discussed the methodology to rank-order features, we will now describe the core of the procedure, namely, the weight computation technique. The basic idea is to assign weights proportional to the magnitude of recent upward trending behavior exhibited by the features. To be more specific, we consider the set of arguments $a_1, a_2 \cdots, a_n$ and assign greater weights to the arguments which have recently indicated high upward trending signal strengths as evidenced by their observed values. We note that this requirement is *a substantial departure* from all extant OWA variants. As mentioned in OWA approach in related work section Eq.4, Yager (Yager, 1993) introduced a set of argument dependent approaches, all of which are unsuitable for the purposes because they all use the argument values themselves to calculate the weights. It is, on the other hand, needed to compute the weights based on the magnitude and direction of change in values. To do this, we borrow a basic idea from Xu (Xu, 2006) which outlines a way to deemphasize the influence of unfair arguments on decision results by weighting these arguments with small values. Based on the intuition developed in Eq.7 and ideas in (Xu, 2006; Yager and Filev, 1999), we have developed a novel deviation based argument dependent approach to determine DOWA weights. Effectively, higher weights are assigned to the arguments which have higher deviation values from historical average deviation values of that argument. We provide the details in the definitions below.

DEFINITION 3.

Degree of Deviation: Let $\mathrm{u}_{\sigma 1}, \mathrm{u}_{\sigma 2,} \dots \dots . . \mathrm{u}_{\sigma \mathrm{n}}$ be a collection of standard deviation values of arguments $a_{\sigma 1}, a_{\sigma 2} \dots . . , a_{\sigma n}$ in the time interval$(t - k)\ to\ (t-1)$. Further, let μ_u be the average value of these standard deviation values, i.e., $\mu_u = \frac{1}{n}\sum_{j=1}^{n} u_{\sigma(j)}$, and $(\sigma_{(1)}, \sigma_{(2)}, \sigma_{(3),} \dots, \sigma_{(n)})$ is permutation of $(1,2, \dots . , n)$ such that $\mathrm{u}_{\sigma(\mathrm{j}-1)} \geq \mathrm{u}_{\sigma(\mathrm{j})}$ for all j=2,…n . Then, it can be called the degree of deviation of the j-th largest standard deviation values (μ_u) as follows.

$$d\left(\mathrm{u}_{\sigma(\mathrm{j})}, \mu_u\right) = 1 + \frac{\mathrm{u}_{\sigma(\mathrm{j})} - \mu_u}{1 + \sum_{j=1}^{n} \left|\mathrm{u}_{\sigma(\mathrm{j})} - \mu_u\right|} (Equation\ 8)$$

DEFINITION 4:

Deviation based Weight*:* *Let* $sdw = (sdw_1\,, sdw_2, \dots, sdw_n)^T$ be the weight vector of the DOWA *operator proposed above, then it is defined* dw_j*as follows,*

$$sdw_j = \frac{d\left(\mathrm{u}_{\sigma(\mathrm{j})}, \mu_u\right)}{\sum_{j=1}^{n} d\left(\mathrm{u}_{\sigma(\mathrm{j})}, \mu_u\right)} (Equation\ 9)$$

Where $d\left(\mathrm{u}_{\sigma(\mathrm{j})}, \mu_u\right)$ is defined by Eq.8.

Properties of DOWA

Based on definitions 2, 3 and 4 above, we assert the following properties for DOWA operator and provide relevant proof of each assertion.

THEOREM 1.

$$DOWA((u_{1,}a_1),(u_{2,}a_2),\dots\dots,(u_{n,}a_n))_t = \frac{\sum_{j=1}^{n} d(\mathrm{u_j},\mu_u)a_j}{\sum_{j=1}^{n} d(\mathrm{u_j},\mu_u)}\ (Equation\ 10)$$

PROOF.

$sdw_j \epsilon\ [0,1]$

$$\sum\nolimits_{\boldsymbol{j=1}}^{\boldsymbol{n}} dw_j = 1$$

$$\therefore \sum\nolimits_{j=1}^{n} d(\mathrm{u_{\sigma(j)}},\mu_u) = \sum\nolimits_{j=1}^{n} d(\mathrm{u}_j,\mu_u)\ \ (Equation\ 11)$$

Based on this Eq. 9 can be rewritten as,

$$sdw_j = \frac{d(u_{\sigma(\mathrm{j})},\mu_u)}{\sum_{j=1}^{n} d(u_{\mathrm{j}},\mu_u)}, j = 1,2,\dots,n\ (Equation\ 12)$$

In this case Eq.7 will be,

$$DOWA((u_{1,}a_1),(u_{2,}a_2),\dots\dots,(u_{n,}a_n))_t = \sum_{j=1}^{n} w_j\, a_{\sigma(j)}$$

$$= \frac{\sum_{j=1}^{n} d(u_{\sigma(\mathrm{j})},\mu_u)a_{\sigma(j)}}{\sum_{j=1}^{n} d(u_{\sigma(\mathrm{j})},\mu_u)}$$

$$= \frac{\sum_{j=1}^{n} d(u_j, \mu_u) a_j}{\sum_{j=1}^{n} d(u_j, \mu_u)} (Equation\ 13)$$

COROLLARY 1. *DOWA is a dependent OWA operator.*

According to the Definition 1 *DOWA* is dependent *OWA* operator because weights (See Eq.9) are function of change in aggregate values.

As for the *orness and dispersion* measures in OWA approach, for DOWA these are defined below.

DEFINITION 5.

Orness:

$$orness_{dowa}(w) = \frac{1}{(n-1)} \frac{\sum_{i=1}^{n} (n-i) d(u_j, \mu_u)}{\sum_{j=1}^{n} d(u_j, \mu_u)} (Equation\ 14)$$

DEFINITION 6.

Dispersion:

$$dispersion_{dowa}(w) = \frac{-\sum_{i=1}^{n} d(u_j, \mu_u) \ln \frac{d(u_j, \mu_u)}{\sum_{j=1}^{n} d(u_j, \mu_u)}}{\sum_{j=1}^{n} d(u_j, \mu_u)} (Equation\ 15)$$

3.4 App Popularity Model

Now we describe the DOWA technique in detail. To reiterate, it aggregates across the different features that determine popularity of an app by dynamically assigning weights to these features based on the magnitude of change in their signal strengths. To apply DOWA in practice, it needs to specify exactly what these features are. In this section, we will describe these features, and, in effect, specify a *model* of app popularity. In the next section, we will perform the experiments to validate the "goodness" of this model.

There is no single convenient measure of app popularity. Rather, there are number of features that we may use for this purpose. It postulates that the following three high level features represent three important signals of app popularity:

(i) *Number of Downloads*: Clearly, higher downloads signal higher popularity (ii) *Number of Active Users*: It turns out that a significant (often large) percentage of users who download an app, end up deleting it. At any given time, *active users* are those who have not only downloaded the app, but still have it on their mobile devices and actively use. Active users are considered one of the strongest signals of user engagement (Amblee and Bui, 2007) – clearly the more the number of active users, the greater an app's popularity. Last, but not least, is (iii) the *store rank* of an app, which is the ordinal display rank the app possesses in its category (in a specific country) in its native store (e.g., the iTunes app store, or Google Play) also referred as NSCR. Out of the above mentioned features *store rank* of a mobile app is publicly available. The number of downloads and the number of active users of apps are not generally available. To capture the essence of these two features, we have designed a set of measurable surrogates: *review valence, user rating* and *user satisfaction.*

Review Valence: Users provide reviews of apps in iTunes and Google play. Currently there exist over one billion native store reviews across 1.4 million apps. We have collected all these reviews for the set of identified apps in our test data set. Prior research (Chevalier and Mayzlin, 2006; Dellarocas, Zhang, and Awad, 2007; Reinstein and Snyder, 2005) has demonstrated strong positive association between user engagement, measured via reviews, to product sales and popularity. Thus, user reviews are used as an indicator of app popularity. In particular, we employ a construct called *review valence* (also referred to as *review polarity*) described in (Asur and Huberman, 2010; Das and Chen, 2007).

$$Review\ Valence = \frac{Number\ of\ positive\ reviews}{Number\ of\ negative\ reviews}$$

To determine positivity and negativity of reviews, several sentiment analysis tools and methodologies such as AlchemyAPI (Alchemyapi.com, 2012), SentiStrength (Thelwall, Buckley, Paltoglou, Cai, and Kappas, 2010), TweetSentiment (Intridea, 2011) and ViralHeat Sentiment analysis API (ViralHeat, 2012) are resorted to assign positive, negative or neutral value for each user review. Among these sentiment analysis tools SentiStrength was chosen due to its superior accuracy. Sentiment analysis tool accuracy statistics is provided in Table 5.

Table 5 - Sentiment Analysis accuracy statistics

Sentiment Analysis Tool	Accuracy
1. AlchemyAPI Sentiment	0.8
2. SentiStrength	0.84
3. TweetSentiment	0.78
4. ViralHeat	0.68

User Rating: Existing research (Amblee and Bui, 2007; Duan et al., 2008; Reinstein and Snyder, 2005) has demonstrated that the number of reviews received for a consumer product is the most active predictor of the size of its installed base. In apps, the situation is a little different, as users have two potential choices: to rate an app (i.e., give it a score between 1 and 5), or, to rate AND review an app. It turns out that a lot of active users simply rate apps without writing an explicit review. Thus, the number of ratings received by an app would appear to be an effective proxy for both downloads and the size of the active user base. Specifically, we define four measurable constructs: (a) *AllVersion Rating Count (AVRCount)*: Total number of ratings received across all versions of an app (b) *AllVersion Rating Score (AVRScore)*: The average rating score received across all versions, (c) *CurrentVersion Rating Count (CVRCount)*: The number of ratings received only for the most recent version of an app, and, (d)*Current Version Rating Score (CVRScore)*: The average rating score received for the most recent version.

One of the key weaknesses of *AVRCount* is that it is age-biased. The longer an app is in the store, its *AVRCount* is likely to be higher. For example, *AVRCount* does not differentiate between an app that has received 1000 ratings in one year vs. another app that has received 1000 ratings in the first week of its release. To include this aspect, *AgedAVRating* is defined as follows.

$$Aged\ AVRating = \frac{AVRScore\ X\ AVRCount}{Age}$$

Here the *AgedAVRating* metric was used as the proxy for downloads and active user base. In addition, another metric was developed based on current version rating as follows.

$$CVRating = CVRScore \times CVRCount$$

Here *CVRating* is used as the indicator of an app's current performance and to estimate its current user engagement.

Table 6 - Features used in Modeling Mobile App Success

Feature	Description
	Base Features
Age	Duration in number of days since the app was released
AVRScore	Average rating across all versions of the app
AVRCount	Cumulative total number of ratings received by an app across all versions
CVRScore	Average rating for current version of the app
CVRCount	Current version Cumulative total number of ratings received by an app
StoreRank	Rank of the app in the Primary category in its native app store
	Derived Features
POSCount	Number of positive reviews received by an app on the date
NEGCount	Number of negative reviews received by an app on the date
ReviewValence	POSCount/ NEGCount
AgedAVRating	AVRScore×AV RCount/Age
CVRating	CVRScore × CVRCount
$\text{UserSatis}_{\text{AV}}$	Count of (3−5) Star Rating / AVRCount
$\text{UserSatis}_{\text{CV}}$	Count of (3−5) Star Rating / CVRCount

User Satisfaction: Clearly, the more satisfied an app's user base, the more likely it is that this app is popular. User satisfaction has been modeled for an app by computing the percentage of positive rating scores (scores ranging from 3-5) across all rating scores (i.e., ranging from1 through 5) awarded to an app. In particular, two metrics are defined, capturing the *all version (AV)* and the *current version (CV)* user satisfaction respectively.

$$UserSatis_{AV} = \frac{Count\ of\ (3-5)\ Star\ Rating}{AVRCount}$$

$$UserSatis_{CV} = \frac{Count\ of\ (3-5)\ Star\ Rating}{CVRCount}$$

Store Rank: Several studies (Chevalier and Mayzlin, 2006; Ghose and Ipeirotis, 2007) have demonstrated that the store rank of a product is a good indicator of success in domains such as books, music and mobile apps. One of the most powerful effects of high rank is increased visibility (e.g., an app ranked 1 in lifestyle will be the first app that will be visible to a user browsing apps in the lifestyle category), and consequently, enhanced popularity. Even though app store ranks are commercially driven (as discussed earlier) they possess both informational content (ranked apps, in general, are more popular) and impact (once an app is ranked high, it definitely gains visibility). Since the approach is dynamic and computes weightage based on signal strength, the error or bias associated with shilling attack or commercial factor will be compensated by the magnitude of the strength assigned for store rank. Hence app store rank is included as one of the features in measuring the overall popularity of an app. The store ranks are directly available for first 1000 apps in each category in Apple iTunes and for the first 480 apps in each category of Google play store. For other apps no store rank data is available.

The features in Table 6 can be categorized into two segments: (i) *Base features*, indicating features directly available from collected raw data and (ii) *derived features* that are computed from the base features. Note that Table 6 presents every feature that could potentially be used to measure mobile app popularity. In practice, only a small subset will be necessary. We will explain as to how this subset is derived in the feature selection component of the DOWA algorithm.

3.5 Experimental Results

To study its effectiveness, DOWA will be subjected to two types of validation tests: (a) comparison against "ground truth" values to test its absolute "goodness" and (b) comparison against a state of the art OWA approach to judge its relative accuracy. In this, (a) presents a problem, as it is not immediately obvious how to avail of ground truth ranks. In other words, we can take a set of apps and DOWA can be used to rank them by popularity, the ***"true"*** ranks of these apps are not generally available to judge the accuracy of the DOWA output. A careful study of the iTunes and Google Play app stores provides an interesting solution – to use *grossing ranks of paid apps*, as the ground truth values. We explain the detail below.

App stores, broadly, present three types of public rank values: free ranks (ranks of free apps), paid ranks (ranks of paid apps) and grossing ranks (apps ranked by how much money the app has yielded). Free and paid ranks cannot serve as a ground truth as they are subject to commercial considerations and are well known to not reflect true popularities, as discussed before. Grossing ranks, on the other hand, are “objective” –if app A is ranked higher than app B, it is known that app A generated more revenue than app B. Therefore, when revenue is used as a proxy for popularity (an often used proxy), it would appear that grossing ranks are a reasonable measure of popularity. However, note that apps can generate revenue in two ways.

- Revenue generated in purchasing the app itself (for paid apps). For instance, if an app was priced at $1.99 and users downloaded it a 1000 times, this app would gross $1990.00.
- Revenue generated by *in-app purchases*, where users pay money in purchasing items as they are using the app. These can range from artifacts in games (such as the “mighty eagle” in the Angry Bird games); to subscriptions of newspapers and magazines (such as getting access to certain paid content in the Wall Street Journal through the WSJ app).

It turns out that apps enabling in-app purchase capability are mostly free apps. Clearly, we cannot use to rank values of these apps (i.e., free, in-app purchase enabled apps) as effective popularity ranks – free app A, enjoying $2000 of in-app revenue, would be ranked higher, but might actually be far less popular than free app B, which made $1500, as the average unit price of in-app items in A might be much greater than that is B. However, for paid apps that have no in-app purchase ability, we can assume that their generated revenue (and, therefore, its grossing rank) is a real surrogate for popularity. Note that paid apps themselves come in various prices, so to be consistent across the ground truth app set, the ranks of paid apps priced at $0.99 is considered. Specifically, two months (From 1st Feb – 31st March 2012) of grossing rank data from the US iTunes app store has been extracted. Out of the top 400 highest grossing apps in each category in this period, it was found that 384 apps had been priced at $0.99 across the categories such as Overall, Games, Entertainment, Lifestyle and Social networking in iTunes on 31st March (and contained no in-app purchase ability). In which 70 are apps from Overall, 37 are from Games, 110 are from Entertainment, 61 are from Lifestyle and the rest 106 apps are from Social networking categories. We use these 384 apps as the ground truth app set and compare their publicly available grossing ranks to the ranks computed by the DOWA technique – we report this later in this section.

Phase 2 Experiments:

To further validate the efficacy of our proposed approach, we use $0.99 priced and in-app purchase not enabled 2718 apps. Chosen apps were ranked as top grossing apps between 1st Jan 2013 to 30 th June 2013. From these, set of apps are chosen on different days and their relevant DOWA and OWA scores are computed. Apps are chosen from the Games, Entertainment, Lifestyle, Social Networking and Overall categories to confirm the consistency of the algorithm's performance. To apply the DOWA technique, for each of these 384 apps, values of features mentioned in Table 6 also have been collected. The feature values were then normalized within $[0 - 1]$.

Error Measurement: We measure accuracy of DOWA using the well-known Mean absolute percentage error (MAPE) metric. For N (N = 384 for Phase 1) apps if we obtain $R_1^0, R_2^0, \dots, R_N^0$ as the real grossing app ranks and $R_1^d, R_2^d, \dots, R_N^d$ as ranks computed by the DOWA approach, then, the MAPE is computed as follows,

$$MAPE = \frac{1}{N} \sum_{i=1}^{N} \frac{\left(R_i^0 - R_i^d\right)}{R_i^0}$$

We describe the details of applying DOWA to the ground truth app set, consisting of two key steps: (i) feature selection, and (ii) rank computation in the following subsection.

Feature Selection

In the App Popularity Model section, we have identified a number of base features that may be used to model popularity of apps (see Table 6), and deserve consideration as input features for DOWA. However, in practice, all these features may not be ideal to serve simultaneously in the input feature set, for various reasons: (a) they may not be independent of each other (i.e., might have auto-correlations), and (b) given a group of correlated features, it is required to select the feature that is “better” than the others. This is referred to as feature selection, and usually has a major impact on the ensuing task.

Table 7 presents the *Pearson correlations* among the features. From this table, we can see that all the selected features are positively correlated among themselves.

Table 7 - Correlation Matrix

Variable Name		1	2	3	4	5	6
CV Rating	1	1.000					
AgedAV Rating	2	0.4605*	1.000				
UserSatisAV	3	0.2626*	0.4430*	1.000			
UserSatisCV	4	0.2114*	0.1384*	0.6250*	1.000		
ReviewV alence	5	0.1211*	0.0459*	0.1539*	0.1448*	1.000	
StoreRank	7	0.0928*	0.1717*	0.0435*	0.0278*	0.4682*	1.000

Correlations marked as* are significant at P < 0:05

The objective of the feature selection step is to decide which of these 6 features need to be considered in the final rank computation. Clearly, it can be generated $\sum_{i=1}^{6} 6_{C_i} = 63$ different feature subset, making an exhaustive test across all 63 practically infeasible. So, we resort an elimination approach based on examining pair-wise correlations. According to statistics, if two features have a correlation greater than 0.4, we will consider them mutually dependent (Fisher et al., 1970). In this case, only one of the two features should be enough to include in the final input feature set.

Table 8 depicts this elimination process. First we consider all 6 features as potential input and calls it set A, as shown in the first row of the table. Next, wenote that $UserSatis_{CV}$ is highly correlated (0.6250 > 0.4) with $UserSatis_{AV}$. So we remove $UserSatis_{CV}$ and derive a second potential input feature set B[22], as

[22] It has been also tested by removing UserSatisAV and keeping UserSatisCV, results were remarkably similar. Hence, in the discussion below when selecting features to discard, it is randomly picked one of over the other, without any loss of accuracy.

indicated in Row 2 of Table 8. Next, we note that $UserSatis_{AV}$ has high correlation (0.4430, i.e. above 0.4) with AgedAVRating, so it removes $UserSatis_{AV}$ from B and derive set C. Following which, we note, CVRating and AgedAVRating are highly correlated (0.4605, i.e. above 0.4), so we eliminate CVRating from set C and derive set D. Lastly, we note among the features in set D, ReviewValence has high correlation with StoreRank (0.4682, i.e. above 0.4), so derived set E by eliminating ReviewValence from the set D. At the end of this elimination step, five distinct feature sets have been derived, namely sets A through E, as possible inputs to the rank computation algorithms.

Taking each of these sets as input, the DOWA output values are computed for each of the 384 ground truth apps and rank the apps according to these values for each day between 19th March and 25th March 2012. The DOWA value of an app on any day is computed based on the previous 7 days' feature value history. The relative grossing ranks for each of these 384 apps for each corresponding day are collected from the iTunes store.

Table 8 - Feature sets for Significance Analysis

Set	Store Rank	Aged AV Rating	Review Valence	CV Rating	UserSatisAV	UserSatisCV
A	X	X	X	X	X	X
B	X	X	X	X	X	
C	X	X	X	X		
D	X	X	X			
E	X	X				

Thus, at this stage, there are two ranks for each app, for each day: a DOWA rank and a True rank (i.e. grossing rank). Using these two ranks, for each day between 19th March and 25th March 2012, the RMSE metric is computed. These are presented for each of the 5 sets A, B, C, D and E for the 7 consecutive days in Figure 6.

Figure 6 reveals that for set A, average RMSE values are relatively high (28%). Errors decrease when using the features in set B (23%) and drop progressively for sets C (15%), and D (10%). When using feature set E however, average RMSE values are greater than those for set D but better than set C. Clearly, set D gives us best possible feature combination. This also demonstrates the sensitivity of the feature selection accuracy and illustrates the importance of feature selection step.

Based on this analysis the mobile app rank model is built using the features in set D, i.e., ReviewValence, AgedAVRating and StoreRank. Note that, we find that using set D, the average RMSE is 12% demonstrating higher accuracy of DOWA approach.

In the next section, wedemonstrate as to how the DOWA approach compares withtraditional OWA.

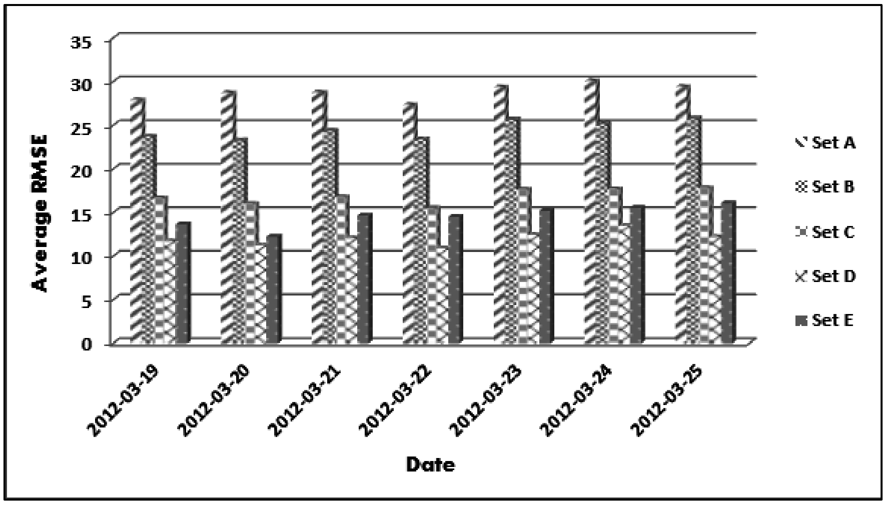

Figure 6 - Average RMSE for Feature Sets

DOWA vs. Traditional OWA

As described earlier, there are a number of ways of computing traditional OWA weights. Here, as a basis of comparison, Yager's PFLQ approach (Yager, 1988) has been incorporated, generally regarded as the premier OWA technique –we will simply refer to this as OWA in the rest of this section. To compare DOWA and OWA, we use first 70 overall category ground truth apps described earlier, compute their ranks using both the DOWA and the OWA approaches, and finally compute the respective RMSE values.

We compare using the computed respective ranks, and the respective RMSE values, OWA and DOWA approaches across ***three*** dimensions.

1. First, the absolute differences of the OWA and DOWA ranks from the real rank are compared

2. Second, the RMSE of OWA and DOWA ranks are compared
3. Third, the RMSE values computed by OWA and DOWA approaches across categories are compared
4. Lastly, the orness and dispersion measurement of OWA and DOWA approaches are compared

The objective of this comparison is to show how different OWA and DOWA approaches are to each other with respect to utilizing all the feature values.

Table 9 tabulates the ranks computed by DOWA and OWA and the true grossing ranks. While we have done this for each of the 384 apps across all the categories, space limitations prevent from showing all– therefore in this table data corresponding to the top 10 grossing apps computed for overall category are presented. In Table 9, for each app, its real iTunes grossing rank (TG Relative Rank), its *DOWA Rank*, its *OWA Rank*, and the absolute differences between its real and computed ranks (Diff-DOWA and Diff-OWA) are presented. It is evident that the DOWA ranks are much closer to the real ranks than the OWA ranks: *The average absolute difference for DOWA (*Diff-DOWA*) is just 2 compared to the average difference of 6 for the OWA case (*Diff-OWA*).*

Table 9 - Experiment Results of Overall Category apps for 19th March 2012

App Name	TG Relative Rank	DOWA Rank	Diff-DOWA	OWA Rank	Diff-OWA
Draw Something by OMGPOP	1	2	1	7	6
NBAJAM by EA SPORTS	2	5	3	10	8
MONOPOLY	3	6	3	13	10
NBA2K12 for iPhone	4	5	1	9	5
UNO	5	9	4	11	6
PicFrame	6	3	3	11	5
Tiger Woods PGA TOUR 12	7	7	0	4	3
Flick Home Run !	8	10	2	2	6
Diptic	9	8	1	3	6
Real Steel	10	12	2	5	5
Average Diff			**2**		**6**

Next, we present in Table 10, MAPE of OWA rank and DOWA ranks.Specifically, the average, median, 90th percentile and standard deviation of RMSE values across ***all*** overall category apps for each day, for DOWA and OWA.

As can be seen from Table 10, DOWA demonstrates substantially lower average, median, 90th percentile and standard deviation of MAPE values in this week compared to OWA case.

Table 10 - MAPE change across days for DWOA and OWA for Overall Category apps

	MAPE -OWA				MAPE -DOWA			
Date	Average	Median	90 Percentile	Std Dev	Average	Median	90 Percentile	Std Dev
19-03-2012	22.92	12.5	19.26	22.89	11.68	9.56	12.3	10.28
20-03-2012	23.44	21.875	21.26	21.71	11.2	8.82	11.6	10.56
21-03-2012	17.19	9.375	22.36	18.04	12.02	8.82	11.8	11.06
22-03-2012	24	15	19.27	21.31	10.86	8.09	11.2	9.94
23-03-2012	24.75	22.5	23.57	17.08	12.41	10.71	11.7	11.01
24-03-2012	24.5	20	22.56	21.33	13.47	11.43	11.1	11
25-03-2012	26.19	25	24.36	21.82	12.16	10	10.8	10.58
Overall	23.28	18.04	21.81	20.6	11.97	9.63	11.5	10.63

We can observe that for DOWA for overall category top grossing apps, average RMSE value is about 12% across all the days compared to 23% in case of OWA – *a 50% reduction in error*. We further observe that the median MAPE value for DOWA approach is 9.6% compared to 18.04% for OWA, exhibiting the same pattern as average MAPE. For DOWA, the 90th percentile RMSE value is 11.5%. This indicates that 90% of the RMSE values are less than or equal to 11.5% for DOWA compared to 21.81% in case of OWA. In summary, *the above result conclusively demonstrates that ranks obtained by the DOWA technique are (a) far more accurate and (b) substantially more consistent, compared to OWA ranks.*

Table 11 - MAPE change across or DWOA and OWA across categories for 384 apps

Category		MAPE -OWA			MAPE -DOWA		
	# of Apps (384)	Average	Median	90 Percentile	Average	Median	90 Percentile
Overall	70	24	15	19.27	10.86	8.09	11.2
Games	37	22.45	17.25	18.68	13.23	11.12	13.36
Entertainment	110	21.35	18.58	20.25	13.75	11.42	12.98
Lifestyle	61	23.48	19.32	19.98	13.82	12.24	13.65
Social Networking	106	21.74	20.97	22.25	13.94	11.29	12.3

To show that DOWA performs well across categories, in Table 11 we present the efficacy of DOWA compared to OWA approach across categories. We have computed Average, Median and 90 percentile RMSE values using both the OWA and DOWA approach for March 22nd across Games, Entertainment, Social networking, Life Style and Overall categories.

Phase 2 Experimental results:

Following table shows the Mean absolute percentile error (MAPE) values for the scores computed using DOWA and OWA approaches compared to Relative Top grossing Rank values for the new set of apps (2718 apps from 1st January 2013 – 30th June 2014).

Table 12 : MAPE change across days for DOWA and OWA approaches

	MAPE -DOWA		MAPE - OWA		
Date	Average	Median	Average	Median	Number of Apps
10-03-2013	15.45	11.25	21.92	17.23	631
11-03-2013	14.07	9.9	20.89	16.78	656
12-03-2013	12.11	8.76	18.96	14.5	696
20-03-2013	11.36	10.12	19.25	15.28	618
Overall	13.24	10.01	20.55	15.94	

In addition to measuring the efficacy of DOWA approach, we evaluated the performance of basic regression and machine learning approaches in predicting the Top Grossing Rank of a given app. For this purpose, first we created two data sets for training and testing purposes. For this purpose same set of 384 apps which were used in phase 1 with 11427 observations (time series data) are chosen. Out of these, 75% of them are used for training purposes and the rest are used for testing purposes (8572 observations).

Using this approach we tried to predict the Top Grossing Rank of an app on time *t* using the same app's previous day values. Following is the prediction formula used in measuring the efficacy of different approaches:

$$TGRank_t = \alpha + \beta_1 Rank_{t-1} + \beta_2 AgedRating_{t-1} + \ \beta_3 Review\ Valence_{t-1} + \ \epsilon$$

Following table (Table 13) shows the summary statistics of the data used in prediction.

Table 13 - Summary Statistics

VariableName	Min.	1st Quarltile	Median	Mean	3rd Quartile	Max
TGRank	-1.892	-0.84	0.04	0	0.806	1.916
Rank	-1.364	-0.809	-0.161	0	0.579	4.742
AgedRating	-4.493	-0.861	0.349	0	0.954	1.559
AVGoodness	-3.501	-0.596	0.206	0	0.842	1.293

Table 14 indicates the record distribution for training and testing for different machine learning models.

Table 14 : Record Distribution for machine learning

Total Number of Observations	11427
Number of Apps	384
Training instances	8572
Testing instances	2855

We have used Support Vector Machine, Random Forest, GLM and Linear regression as the methodologies to predict the $TGRank_t$. Table 15 shows the different set of parameters used in Random Forest and SVM.

Table 15 - Parameter Values

Methodology	Parameter name	Value
SVM Regression	type	eps-regression
SVM Regression	kernel	radial
SVM Regression	cost	150
SVM Regression	gamma	0.001
SVM Regression	epsilon	0.1
SVM Regression	# of support vectors	9400
Random Forest	ntree	500

Following table 16 shows the MAPE values of each approach we have used in predicting the Top grossing rank of apps. As can be seen from table, GLM and

Linear regression gives higher MAPE values than the other two approaches. MAPE values for Random Forest and SVM are still much greater than the MAPE values by DOWA. Therefore, it can be concluded that DOWA is much effective than the other approaches like OWA, regression and machine learning based models.

Table 16 - Machine learning results

Method	**MAPE**
GLM Regression	0.644
Linear Regression	0.644
Random Forest	0.451
SVM Regression	0.572

Lastly, OWA and DOWA approaches we have compared in terms of their orness and dispersion values to judge whether the DOWA approach significantly deviates from the core objective of the OWA class of approaches in terms of weighting various feature values. The orness and dispersion measurement for each app for each day between 19th March 2012 and 25*th* March 2012 were computed. The orness values are between 0 and 1, where orness value of greater than 0.5 indicates it is giving more weight to the features with higher absolute values. So in Table 17, we present the minimum orness values across all apps in each day. We can observe that the orness values for DOWA approach are in the same range as that of OWA approach. Next, we have computed the dispersion across 70 apps for each day and reported the minimum dispersion for each day in Table 17. We can observe that minimum dispersion values for both OWA and

DOWA approaches are in the same range across the target 7 days. *Thus we can be conclude that both DOWA and OWA approach has similar characteristics in terms of weight distribution across feature values.*

Table 17 - Orness-Dispersion Measures

	Min-Orness		Min-Disperson	
Date	OWA	DOWA	OWA	DOWA
19-03-2012	0.774	0.745	0.842	0.812
20-03-2012	0.778	0.764	0.856	0.852
21-03-2012	0.796	0.784	0.845	0.862
22-03-2012	0.789	0.779	0.835	0.854
23-03-2012	0.795	0.824	0.825	0.867
24-03-2012	0.805	0.814	0.855	0.843
25-03-2012	0.785	0.754	0.865	0.879

3.6 Discussion and Conclusion

In this chapter we have addressed the practically relevant problem of computing popularity ranks of mobile apps. This problem is made difficult owing to the commercial factors and dynamic nature of the features signals that impact popularity and renders existing rank computation approach unsuitable. To solve this, we have effected a substantial extension to Yager’s well-known OWA approach by relaxing the condition that feature weights need to be static. In particular, we have introduced a new way of weight computation based on change in signal strengths of the features and re-ordering weights based on current relative signal strengths. The approach, termed DOWA, was validated for accuracy against established ground truth values as well as against state-of-the-art OWA methods. The results were very encouraging: in an absolute sense, DOWA was consistently within 10% of ground truth values, exhibiting high absolute accuracy of DOWA. The DOWA grossly outperforms OWA, beating the latter by at least a factor of 2:1. When compared to PFLQ, the premier extant OWA technique, it performed far better – PFLQ was at least 100% as inaccurate along every measure studied.

This study is enriched with important theoretical contributions. First, the novel DOWA approach discussed in this study has important theoretical contribution in the research of multi-criteria decision making techniques in general and Ordered Weighted Average techniques in particular. Second, the proposed ranking mechanism for mobile apps extends the research on mobile apps.

This study also makes several significant contributions to important constituents of app eco system. First, the ***true*** app rank can help consumers and advertisers who are constantly faced with choice dilemma. It would be vital for advertisers knowing the app popularity ranks in real time and programmatic buying of this information would increase the effectiveness of mobile advertising. When there is an Ad Request comes to Supply side platforms or RTB Exchanges or Demand Side platforms as a third party platforms which can provide the real time information about popularity of mobile apps or recommend the apps which are popular at that time. For example *"An agency media planner, managing the launch campaign for an expensive running shoe for women, does not know precisely which top 50 mobile apps offer the best reach into wealthy female fitness lovers. Not only are these metrics likely to change in the near future, it is* *also probable that new and cool fitness apps will emerge to challenge the top 50",* thus in this kind of scenarios it is always better programmatically buy the app inventory (i.e. buy the app inventories in real time which are popular among female users based on the different signals apps have received at time *t).*

Secondly, app stores can readily use this ranking mechanism to overcome the issues of existing ranking system. Thirdly, for developers and publishers the proposed ranking approach can help in self-assessment and competitive analysis in order to survive in the hyper-competitive apps market.

The practical impact of DOWA goes beyond app rank computation. It represents a robust technique usable in any scenario possessing similar characteristics, namely dynamically varying signal strengths. Application areas include computing relative popularities of many consumer products, including online music, books, etc. Finally, such investigations can have rich potential to extend Information Systems research.

Chapter 4
Study III: Programmatic media acquisition based on audience profile

4.1 Background and Motivation

As discussed earlier in Chapter 1, during the period from December 2011 to December 2012 the average time spent on smartphones by a US consumer has increased from 94 minutes to 127 minutes (i.e. by 35%) (Simon, 2013), while the average time spent on web has decreased by 2.4% (i.e.72 minutes to 70 minutes). On average US consumers are spending 1.8 times more on apps compared to the web (Simon, 2013). Statistics indicate that roughly 224 million people use mobile apps on a monthly basis, compared to 221 million desktop users i.e. mobile app users are slightly more than desktop users (Mary, 2013). Moreover, it has been observed that mobile have become the first screen and made TV as the second screen during the recent super bowl event[23]. This indicates that brand owners need to concentrate more on mobile advertising in order to reach more customers. Thus, mobile apps have become a lucrative media with a growing customer base and promising revenue.

With the growing customer base, understanding audience properties is crucial to yield greater business value for mobile advertisers. However, audience tracking is

[23] http://blog.flurry.com/bid/93898/The-Screen-Bowl-Mobile-Apps-Take-On-TV

far more difficult in mobile context. Commercial audience measurement agencies Neilson, ComScore and Quantcast determine the audience characteristics of media (such as print, radio, TV and internet) often using panel based approaches. In this approach, set of users with known demographic information are recruited, and their behavior is captured either by survey or by instrumenting their gateway devices (cable box and browser). Then demographic attributes of these users are extrapolated to wider audience. In addition, behavioral weights are also used to correct for potential biases in the recruited panel. This approach leads to a reliable audience estimates as the popularity of TV shows and websites are persistent for quite a long time (i.e. at least for months). So the real-time collection of demographics for TV shows and web sites is less of an issue. For example, a popular website such as CNN.com is unlikely to be wiped out of the map in 60 days. Similarly, popular TV show American Idol is likely to be popular at least for 90 days. In other words, popular websites and regular TV shows hardly demonstrate churn. However, unlike the traditional media (such as TV and web), mobile app popularities are highly transient. Table 18 illustrates the top 5 popular apps based on their store ranks on 1st of May and 1st of June 2013 in United States under Games Category. Further, panel app based audience measurement is not a cost effective solution . For example Singapore' leading Telecom spends around S$ 300 K / year on their panel for measuring the audience for their internal apps (~ 15 apps). Further, users concerned abouth their privacy and security especially for mobile based panel techniques. In addition, since there are too many apps available in the market due to low production cost (~ $ 6,453), it is very difficult to measure the each apps' audience information using panel based approach. In contrast, the average broadcast network drama in the US costs $3 million an episode to produce (Carter 2010.)

Table 18 - Top 5 Games Apps in US Store

No	Top 5 Apps for: iPhone - US Games Category on 1st May 2013		Top 5 Apps for: iPhone - US Games Category on 1st June 2013	
	Free	Paid	Free	Paid
1	Robot Unicorn Attack 2	Survivalcraft	Dumb Ways to Die	Heads Up!
2	Draw Something 2™ Free	Cut the Rope: Time Travel	Candy Crush Saga	Bloons TD 5
3	PAC-MAN DASH!	Minecraft – Pocket Edition	Tetris® Blitz	Block Fortress
4	Iron Man 3 - The Official Game	Draw Something 2™	Snoopy Coaster	Plague Inc.
5	Whats The Movie?	Teenage Mutant Ninja Turtles: Rooftop Run	Fast & Furious 6: The Game	Kick the Buddy: No Mercy

As can be seen from Table 18, paid and free apps that were popular on 1st of May 2013 were no more popular on 1st of June 2013 (i.e. within 1 month/31days period). Thus, it can be inferred that mobile app popularities are not persistent. Considering the top 100 apps, on average 46% churn over 30 days and 85% churn in 90 days[24]. Interestingly churn rate of games and lifestyle apps are extremely high (80% - 90%). When one wants to try the panel based measurement in this scenario, the process of panel based data collection needs to happen almost every week or even every day to have an accurate measurement, which is impossible to

[24] http://blog.flurry.com/bid/90743/App-Engagement-The-Matrix-Reloaded

carry out. In summary, app popularities are highly volatile and transient in nature and therefore traditional panel based techniques cannot be used in measuring the app audience.

With this backdrop, we aim to resolve this challenge by proposing a non-panel based reliable scientific technique. We propose a hybrid approach based on classification and prediction. In the classification, each app would be assigned to one or multiple fine grained classes. Based on the class to which the app is assigned, relevant demographic such as Age, Has Children & Education will be assigned to the app (i.e. Classification and mapping approach). App's gender would be predicted using the machine learning based prediction approach.

The proposed hybrid approach has several advantages compared to the traditional panel based approach. First, the approach is scalable with the increased number of mobile apps (currently 1.4 million within Android Playstore and Apple iTunes). Second, audience demography of new apps can be instantly computed as the apps get added to the app store and become popular, without waiting for the panel to be recruited.

The findings of this study could yield significant contributions to important constituents of the app eco system (which is one of the fastest growing e-business of the decade). The proposed audience measurement mechanism can help both mobile ad exchanges and app tech agencies to target specific consumers using the programmatic buying techniques. Further, by using the audience estimation method, app developers and platform owners (e.g. Apple, Android) could reach more consumers and yield greater profit. Being able to measure the audience would increase the reach and visibility of the app. Moreover, audience measurement can also help consumers to identify the most suitable app which can fulfill their need.

We structure the rest of this chapter as follows. In the immediately following section, weprovide the related literature for the approach. Then the proposed solution is detailed and followed by results. Finally we discussed the practical, research implications and future work.

4.2 Related Work

We discuss briefly the literature and methodologies related to audience measurement and online advertising strategies in this section.

4.2.1 Audience Measurement

Prior research has studied demographic attribute prediction using user's web usage pattern. Particularly, previous studies have used content of the websites (Kabbur, Han, and Karypis, 2010), various types of internet user statistics such as web page click though data (Hu et al., 2007), search term (Murray and Durrell, 2000; Zhang et al., 2006) to derive user demographic attributes. Adar (2007) predicted the demographic information of online audience using vector comparison (known vs. unknown users) and a bias value for web pages. Hu et al., (2007) used several methods including Bayesian classification model, similarity between users, and multiple classifiers to predict demographic attributes of users. Murray and Durrell, (2000) analyzed the search terms entered and web pages accessed by users and predicted user demographic attributes using Latent semantic analysis (LSA).

In practice, cookies are commonly used to gather long term data of individual browsing histories. Cookie is a piece of text sent from website and stored in a user's web browser while user is browsing a website. When the user browses the same website again in future, the cookie is sent back to the website to notify web user's previous activity. Despite of the popularity of cookies, they are often criticized for privacy concerns (Mayer-Schönberger, 1998). Internet marketing research agency ComScore, measures the web audience, using a tag that is propagated throughout the website to be tracked, which in turn will measure traffic, page views and other related information. To measure audience attributes ComScore regularly maintains around 2 million panelists who have installed a background monitoring software that tracks their online behavior. In addition, series of weight adjustments are carried out to generate accurate country specific (e.g. US) or global web demographic. This is detailed by comScore as *"Demographic information is gathered from our panel. When someone opts into the comScore panel, they are required to fill out a short questionnaire where we gather demographic information for themselves as well as other people in the House Holds who will be using the metered computer. We then use census population estimates to project out to the total internet population".* Similarly

Quantcast, a web analytics service, measures the web audience statistics by allowing the registered sites to run its data collection feeds, web beacons and anonymous cookies to track the online behavior of web users. Based on the online behavior of each user, Quantcast builds a profile of that person's browsing habits and hence extrapolate demographics.

The literature on user demographic prediction provides with the basic understanding on audience estimation. However the approaches used in literature cannot be utilized for mobile apps for several reasons. First as discussed earlier, due to the rapidly changing (or volatile) popularity of apps and continuous additions of new apps, the panel based approach will not work for mobile apps. Second, due to the huge number of apps available in the market (1.4 million for Google Play Store and iTunes stores), recruitment of panels for measuring demographics is an impossible task. Third, similar to cookies, mobile app based cookie tracking such as Safari flip-flop, HTML5 first party cookies and UDID (unique device identifier) have also been criticized for privacy concerns and apps with these tracking tools have been rejected by platform owners. Therefore, in order to measure the audience for mobile apps, this study explores a novel, non-panel based techniques that does not invade the privacy of users.

4.2.2 Mobile Advertising

This subsection reviews some of the prior literature in the space of online advertising strategies, which involve the web and mobile advertising domains.

There has been extensive research into understanding the new media channels for advertising products, namely the Internet in general and more recently social media platforms. A widely popular book by Robbin Zeff and Bradley Aronson brilliantly summarizes the successful internet ad models, strategies to doing good market research on the internet, the many types of ad management tools, ad trading strategies and the policy and legal aspects of internet advertising. A fair amount of research has gone into understanding audience measurement in an online setting from a user behavioral point of view (Danaher and Mullarkey, 2003; Goldfarb and Tucker, 2011; Yan et al., 2009). The advertiser's perspective has also been studied by a few researchers who have looked into the economic value of advertising, the different advertising strategies, the consumer demand functions and the marketing and sales impacts of different ad types (Bagwell, 2007; Chintagunta and Vilcassim, 1992; Erickson, 1992; Johnson and Myatt, 2006). Over the years, researchers have also shown an increasing interest in studying the dynamically traded online ads. A famous working paper by Edelman and Schwarz (2011) talks about the Generalized Second Price (GSP) auctioning of ads and its role in internet advertising. Along this line, studies have rigorously investigated optimized ad bidding strategies for competing firms, the differences between online and offline auctioning of advertisements, the game theoretic aspects of the auctioning process and the various yield optimization models (Borgs et al., 2007; Cary, et al., 2007; Ghosh et al., 2009; Massad and Tucker, 2000). Needless to say, advertising in new age media has opened up several new research avenues for interested researchers in reference disciplines spanning computer science, economics, media design and the social sciences to name a few (Cho and Khang, 2006; Muthukrishnan, 2008, 2009)

Compared to this vast collection of scholarly literature published in the space of internet advertising, the amount of studies focusing on the more recent channels of advertising like the mobile phone is relatively scarce. Firstly, there have been a few empirical studies looking into customer responses to mobile phone ads and the antecedents influencing consumer reactions to such ads (Leppäniemi and Karjaluoto, 2005; Tsang, Ho, S.C. and Liang, 2004). Secondly, a few notable studies have also looked at the technical infrastructure and framework required for mobile phone ads to succeed (Aalto, Gothlin, Korhonen, and Ojala, 2004; Varshney and Vetter, 2002). Thirdly, there have been noted efforts that look into the economic implications and business opportunities stemming from delivering ads to consumers over mobile phones (Sharma et al., 2009; Komulainen et al., 2006). Interestingly, we noticed a few commonalities across the various papers which have been reviewed in the above three categories. First, most empirical studies rely on text messaging as a medium for ad transmission and second, most studies use frameworks that are essentially user-focused, probing particularly the attitudinal and behavioral aspects of ad viewing. Certainly, there is lack of literature focusing on ad marketing from the perspective of the ad platform owners (i.e. publishers, advertisers, ad networks etc.). Further, to the best of knowledge there has not been any systematic research that looks specifically into the domain of in-app ad serving for smart phones, which is a rapidly growing advertising segment in the present times. We focus on addressing these research gaps.

4.3 Proposed Solution Approach

We detail the intuition behind the proposed approach in this section.

4.3.1 Intuition

Audience demographics are the quantifiable measures of a given population. Audience demographic data are used widely in public opinion polling, marketing and advertising. Generally, demographic data of a person include gender, age, ethnicity, income, language and even location. Precise estimation of audience demographics can help in targeting the right audience through the media (such as web, mobile, TV, Radio etc.). Interactive Advertising Bureau (IAB) (IAB, 2011), an organization for developing industry standards for advertisements has proposed a standardized taxonomy for classifying mobile apps, based on the advice received from taxonomy experts. This IAB taxonomy has 23 broad categories in Tier-1, 371 sub-categories in Tier-2 and infinite number of categories in Tier-3. Table 19 shows some of the IAB's Tie-1 to Tier-2 category mapping. We intend to measure some of the audience properties of mobile apps in two ways in this study. Firstly, by classifying apps into IAB defined Tier-2 categories and then derive the specific audience properties ("Age", "Has Children" and "Education") of each app using the category-audience mapping. Secondly the gender distribution of each has been predicted using machine learning approaches. Both the approaches are detailed below.

Table 19 - IAB Tier-1 to Tier-2 mapping

Tier -1	**Business**	**Family & Parenting**	**Sports**	**Society**
Tier -2	Advertising Agriculture Construction Government Human-Resources Marketing	Adoption Babies and Toddlers Daycare/Pre School Family Internet Pregnancy Special Needs Kids	Auto Racing Base Ball Bicycling Cricket Football Inline Skating Olympics Swimming	Dating Divorce Support Gay Life Marriage Senior Living Teens Weddings Ethnic Specific

Ideally the first goal is to *generate top n categories for a given app A and estimate the set of app audience demographic properties based on the "category – audience demographic" mapping*. For example, we could estimate the audience of iTunes app 'Brides', which is placed under 'Lifestyle' category in the Apple iTunes store. First the app ('Brides') would be classified into a set of IAB Tier-2 categories. In this case, the iTunes app 'Brides' will be classified into IAB categories such as 'Society: Weddings', 'Society: Marriage', 'Style & Fashion: Beauty', 'Style & Fashion: Fashion' and 'Hobbies and Interests: Photography'. In addition to the classification it has also obtained a class membership score for the app in each of these categories. For example for app (`Brides') it receives 0.4 as the score corresponding to `Society: Weddings' category, 0.3 to `Society: Marriage' category, 0.2 for `Style & Fashion: Beauty' category, 0.05 for `Style &

Fashion: Fashion' and 0.05 for `Hobbies and Interests: Photography'. The second step involves creating the demographic against each of the IAB Tier-2 categories. For this, first a set of apps is identified in each category, whose demographics are well known. For example, it is known that slot machines are used by older female groups. So it can be assigned similar demographics to the category related to slot machine. There are several ways one can obtain the demographics of such an app. These apps are called reference apps. Having identified multiple such reference apps, and their corresponding demographics, for a given category, the demographics of corresponding reference apps is consolidated and overall demographics of the category are derived. Thus, for the given app 'Brides', using the relevant category membership, the audience demographics would be estimated as age = '20-35', 'education = 'grad school & above', 'having children = no'.

The second goal is to *for a given app A predict its gender demographic distribution.* For example, for the same app mentioned above (App "Brides"), relevant gender distribution would be 20-80. Meaning that 20% of users could be male and rest 80 % would be female users. We observed that deriving the gender distribution of an app using the classification approach discussed above did not yield satisfactory results. This is may be because of it is difficult to predict the gender information at the category level, but other attributes like Age can be predicted at the category level. For example, "Angry Birds Rio" app is placed under "Games Arcade" category in Google play store. Using the category information, we can only infer the potential user age group as Teen, GenY and

Middle Age. At the same time it is difficult to estimate the Gender distribution of using only the category information. Thus we propose using machine learning approach and predicting the gender can achieve better accuracy.

Having described the intuition and the high level approach, we describe the details of the solutions in the next section.

4.3.2 Solution Details

The solution has four major components: (1) category-demographic mapping, (2) app classification, (3) audience measurement (Age, Education and Has Children) and (4) gender prediction. Each component is described in detail.

I. Category Demographic Mapping

As described before, it is relied on IAB Tier-2 category for the demographic identification of an app. One of the important steps in the approach is determining the demographic of each IAB Tier-2 category. For this purpose, a set of reference apps was identified for each IAB category. Reference apps are apps that have corresponding websites or Facebook fan-pages, where the audience demographics are known. For example, for IAB Tier-2 category "Travel: Hotels" has been identified apps like "Hotels.com", "Travelocity - Book Hotels, Flights & Cars"and "Kayak" which have their respective sister websites such as hotels.com, travelocity.com and kayak.com. In addition, this set of apps have their respective Facebook pages as well (e.g. www.facebook.com/travelocity). In the proposed approach of demographic identification of mobile apps, we assume that mobile app user demographics are approximately similar to the user demographics of their corresponding sister websites or relevant social media pages (e.g. Facebook fan-pages). We have validated this by calculating the semantic similarity between reference app web site contents and description of randomly selected apps in each category. Using this assumption it was combined with the demographics of these sister websites from known sources (Alexa, 2012; Quantcast, 2013) and Facebook fan-pages to derive the demographics of each reference apps. Next, using the demographics of reference apps of each IAB category, we derived the demographic of each IAB Tier-2 category. As an example, Table 20 shows the demographics of the “Society: Weddings” category was derived using the above mentioned approach.

Table 20 - IAB Category Demographic Mapping

Tier - 2 Category	***Demographics***

Society: Weddings	Age {Child: 0%, Teen: 10%, GenY: 75%, Middle Age: 10% & Old: 5%} Gender {Male: 30% & Female: 70%}
	Has Children {Yes: 15% & No:85%}
	Education: {No College: 5%, College:30% & Grad School: 65%}

II. App Classification

a. Data Preprocessing and Feature Extraction

Once the demographic for each IAB Tier-2 category has been identified, the process of demographic identification of an app involves identifying the best possible (in this case it was top 5) IAB Tier -2 categories to which an app can belong to. This process classifies the existing apps into identified IAB categories. For this purpose publicly available app description were used as the main source. For classification it was followed a supervised machine learning approach. Initially to train the classifiers approximately 25 apps in each category has been manually identified. Hence, for all 371 IAB Tier-2 categories, 9205 apps as the training set have been identified. All these apps we have validated twice for the accuracy of categorization into their respective classes (or categories) by professional lexicographers. For example, under the category "Home & Garden: Gardening", apps such as "Garden Insects", "Gardening and Landscape Guide" and "Vegetable Gardening Guide" are identified as training instances.

For each app in the training set, a set of features has been identified. Figure 7 shows the details of the feature extraction process that includes several steps.

Step1: In step 1 each app description is checked for special characters (see Table 21) and if found, it will be removed from the app description, then it is subjected to language test. We have identified several non-English apps in the training of samples. There were many country specific IAB categories under the

Tier 1 category "Travel" and we observed that some of the apps were not in English. For example, for the Tier 2 category "Travel: Saudi Arabia" apps such as "Riyadh Food - الرياض مطاعم" and "Al Tayyar Travel - للسفر الطيار" were identified as training instances. As the app description was written using both English and Arabic, a translator package (i.e. Bing Translator –Microsoft Corporation, 2011) has been included to handle app descriptions that contained languages other than English. Then all the stop words have been removed (see Table 21) from the app description.

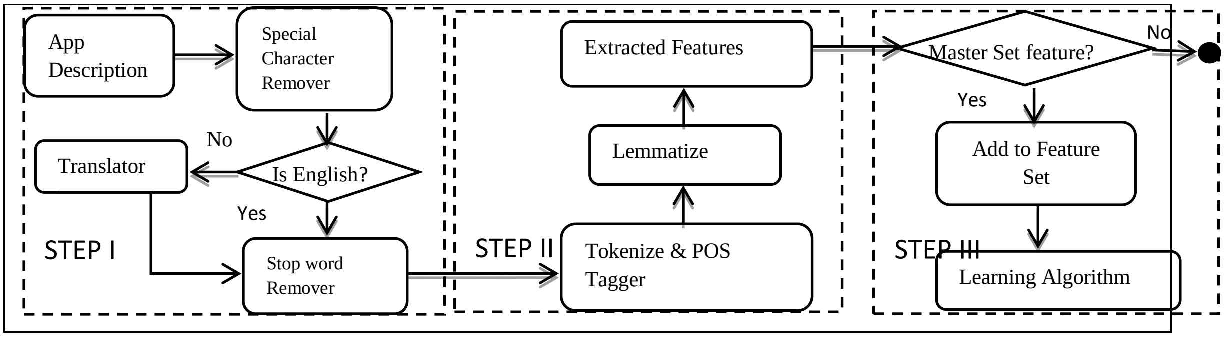

Figure 7 - Feature Extraction Process

Step 2: In step 2, the processed description was subjected to part of speech tagging and lemmatization. The Stanford part-of-speech tagger is used (Toutanova, Klein, Manning, and Singer, 2003) to attach a part-of-speech tag to each token (i.e. Word) in the app description. More precisely, the app description is parsed into sentences, which are then processed by the part-of-speech tagger. When supplied with a sentence, the tagger can produce an ordered list of part-of-speeches as the output for each word in the sentence (such as noun, verb, adjective, etc.). For example, the app called “Beer Calculator” had the sentence like the following in its description: “*By now we all know that alcohol is bad for you, yet most of the people will still go out to have a beer*”. When this sentence is subject to part-of-speech-tagger the word ‘By’ was tagged as a preposition, ‘now’ as adverb, ‘we’ as personal pronoun and ‘all’ as a determiner, and so on. Thus the overall tagging results would be By/IN now/RB we/PRP all/DT know/VBP that/IN alcohol/NN is/VBZ bad/JJ for/IN you/PRP,yet/RB most/JJS of/IN will/MD still/RB go/VB out/RP to/TO have/VB a/DT beer/NN, where IN, RB, PRP, DT, VBP, NN,VBZ, JJ, MD stands for preposition, adverb, personal pronoun, determiner, Verb, Noun, adverb, Verb, adjective and model respectively. Once the descriptions were tagged, only the verb, adverbs and nouns were extracted as the initial features. Then extracted features were subjected to lemmatization in order to get the root word (e.g. “running” would be lemmatized as “run”) form a particular extracted token.

Step 3: Once the initial set of features were extracted based on the above mentioned procedure, in step 3, it was subjected to master feature set check. The master feature set is a bag of words that contain words related to app domain. The initial master feature set was created by lexicographers based on the bag of words (i.e. dictionary) related to app domain.

To build the master feature list, a corpus for each category has been created by taking a sample of 100 apps per category and then came up with higher frequency and higher *IDF* (i.e. rare words) tokens for each category (top 100 tokens). Then it was added the tokens into the initial master feature list. If the extracted top word appears in the master feature set, then it will be considered as one of the feature for a given app. Thus, for each app selected for the training, features were extracted and kept in a file in the following format.

"<feature1> <feature2><feature 3>………………………… <feature_n>".

Now we have extracted the features and, next proceed with building the classification model.

Table 21 - Stop words & Special Characters

Stop words (Only some)	"a", "about", "above", "above", "across", "after", "afterwards", "again", "against", "all", "almost", "alone", "along", "already"
Special Characters	(¬》 《㋡〉 ⅰ 〠⎫_✹★◆✦✔•■®♥#*■ν◇▲▼□□– ⅱ ⅲ、'～,àè:)ññé@#$%&(ñó#/.!;-=üä?【】ⅳ」「)

b. Building classification model

Multinomial Naïve Bayes, TF-IDF and Support vector machines are used as the initial classification approaches in classifying the apps into the possible IAB Tier-2 categories. A brief introduction about these methodologies are detailed below.

Naïve Bayes:

Since the training input is pre-processed app description, token-based naive Bayes classifier is used to compute the joint token count in app description and category probabilities by factoring the joint into the marginal probability of a category times the conditional probability of the tokens given the category defined as follows.

$$p(tokens, cat) = p(tokens|cat) * p(cat)$$

Conditional probabilities of a category given tokens are derived by applying Bayes's rule to invert the probability calculation:

$$p(cat|tokens) = p(tokens, cat) / p(tokens)$$

$$= p(tokens|cat) * p(cat) / p(tokens)$$

Since Naïve Bayes assumes that tokens are independent of each other (this is the "naive" step):

$$p(tokens|cat) = p(tokens[0]|cat) * \ldots * p(tokens[tokens.length - 1]|cat)$$

$$= \Pi_{i < tokens.length} p(tokens[i]|cat)$$

Then, using the marginalization the marginal distribution of tokens has been computed as follows:

$$p(tokens) \; = \; \Sigma cat' \, p(tokens, cat')$$

$$= \; \Sigma cat' \, p(tokens|cat') \; * \; p(cat')$$

In addition, maximum a posterior (MAP) estimate of the multinomial distributions also calculated for $p \; (cat)$ over the set of categories, and for each categorycat, the multinomial distribution $p(token|cat)$ over the set of tokens. Further, it has been employed the Dirichlet conjugate prior for multinomials, which is straightforward to compute by adding a fixed "prior count" to each count in the training data. This lends the traditional name "additive smoothing". After building the Naïve Bayes classifier, extracted features with the respective categories are passed as the input to build the classification model.

TF-IDF:

This classifier is based on the relevance feedback algorithm originally proposed by Rocchio (Rocchio, 1971) for the vector space retrieval model (Salton and McGill, 1986). In TF-IDF, it has been considered the app description of each app as the input document which can be classified into many IAB categories. In other words TF-IDF classifier was adopted to find the best matching category for the given app description. Thus, TF-IDF approach captures the relevancy among words, text documents and particular categories. TF-IDF for a given word extracted in the Step 1 was computed using the following formula:

$tfidf(w_i) = f_D(w_i) \times \left(log\left(\frac{|D|}{df(w_i)}\right)\right)$, where $|D|$ is the total numberof documents in the corpus and $f_D(w_i)$ is number of times word w_i appears in a given document d. This word weighting heuristic says that a word w_i is an important indexing term for document d if it occurs frequently in it (i.e. the term frequency is high). On the other hand, words which occur in many documents are rated less important indexing terms due to their low inverse document frequency. Training the classifier is achieved by combining document vectors into a prototype vector $\underset{c_j}{\rightarrow}$for each class C_j. First, both the normalized document vectors of the set of app description for a class (i.e. positive examples) as well as those of the other app descriptions for the other classes (i.e. negative examples) are summed up. The prototype vector is then calculated as a weighted difference of each.

$$\underset{c_j}{\rightarrow} = \alpha \frac{1}{|C_j|} \sum_{\underset{d}{\rightarrow} \epsilon\, C_j} \frac{\vec{d}}{||\underset{d}{\rightarrow}||} - \beta \frac{1}{|D - C_j|} \sum_{\underset{d}{\rightarrow} \epsilon\, D-C_j} \frac{\vec{d}}{\left\|\underset{d}{\rightarrow}\right\|} - (Equation\ 16)$$

α and β are the parameters that adjust the relative impact of positive and negative training examples. C_j is the set of training documents assigned to class j and $||\underset{d}{\rightarrow}||$denotes the Euclidian length of a vector $\underset{d}{\rightarrow}$. Learned model for each class is represented by resulting set of prototype vectors (see equation 16). This model can be used to classify a new document d'. Again the new document can be represented as a vector $\underset{d'}{\rightarrow}$ using the scheme described above. To classify d' the cosines of the prototype vectors $C_j$with $\overrightarrow{d'}$ are calculated. Finally class for the document d'would be assigned based on the highest document vector cosine score.

$$H_{TF-IDF}(d') = \; argmax_{C_j \varepsilon \, C} \cos\,(\overrightarrow{C_j}, \overrightarrow{d'})$$

In this way, TF-IDF classifier has been trained using the training data of 9205 apps. Then the trained model is used to predict the class for the rest of the apps.

Support Vector Machine:

Support Vector Machine (SVM) is a supervised learning algorithm developed over the past decade by Vapnik and others (Joachims, 1998; Vapnik, 1999). The algorithm addresses the general problem of learning to discriminate between positive and negative members of a given class of n-dimensional vectors. The SVM algorithm operates by mapping the given training set into a possibly high-dimensional feature space and attempting to locate in that space a plane that separates the positive from the negative examples. SVM Multiclass library (Joachims, 2008) has been used to train the SVM classifier which uses the multi-class formulation described in (Crammer and Singer, 2002). This formula has been optimized with an algorithm which makes it more scalable in the linear case. SVM Multiclass library expects the training and testing data in the following format.

<line> .=. <target> <feature>:<value> <feature>:<value> ... <feature>:<value> # <info>

<target> .=. <integer>

<feature> .=. <integer>

<value> .=. <float>

<info> .=. <string>

Here target and feature should be represented by integer. Thus, all the 371 categories have been given a unique identifier from 1-371 and each unique feature is assigned a unique number across training and testing data.

The target value and each of the feature/value pairs are separated by a space character. Feature/value pairs are ordered by increasing feature number. Features with value zero are skipped in building the model. The target value denotes the class of the example via a positive (non-zero) integer. So, for example, the line

6 1:0.42 3:0.34 9284:0.2 # angry birds

specifies an example of class 6 which is for game for which feature number 1 has the value of 0.42, feature number 3 has the value of 0.34, feature number 9284 has the value of 0.2, and all the other features have the value of 0. In addition, the app name “angry birds” is stored with the vector, which can serve as a way of providing additional information when adding user defined kernels. All the features are represented by respective tf-idf values for each category.

As mentioned above, all three classifiers are trained using the same training data with the different representation.

We describe the audience measurement process using the output of classification process in the section following.

III. Audience measurement

Once an app has been classified using the previously described approach, now we describe as to how it is assigned the demographic to each app. Assume an app A is classified into a set of categories $\mathrm{c}_1, \mathrm{c}_2, \ldots. \mathrm{c}_\mathrm{n}$ with their respective classification scores $s_1, s_2, \ldots. s_n$. Then their respective weighted average scores are calculated ($ws_1, ws_2, \ldots, ws_n$). These weighted average scores are required, since chosen classifiers return score values in different ranges and more importance should be given to the category which has returned the highest score. If one assumes $s_1 > s_2 > \cdots > s_n$ then the ws_i could be calculated using *Proportional Fuzzy Linguistic Quantifier* (PFLQ) technique proposed by Yager (1988) as follows;

$$ws_i = \frac{{s_j}^{\alpha}}{\sum_{j=1}^{n} {s_j}^{\alpha}} \ where \ \alpha \ \epsilon \ (-\infty, +\infty)$$

After calculating weighted average scores for each category, the overall demographics of app A is estimated as, here D_i is the consolidated demographics for the category c_i. Further D_i is a $(1 * n)$ matrix and *n* is the number of different demographic dimensions. Using this approach “age”, “education” and “has kid” demographics is calculated. Here we have used the $\alpha = 0.1$ for which accuracy was better.

$$D(A) = \sum_{i=1}^{n} ws_i * D_i$$

IV. Gender Prediction

Estimating the Gender distribution of mobile app users using the above mentioned classification approach did not yield better accuracy compared to other metrics such as “Age”, “Education” and “Has Kids.” It has been observed that IAB Tier-2 categories cannot be used to estimate the relevant gender distribution of an app which belongs to more than a category. Thus we have employed text mining and machine learning based approaches to predict the gender distribution of mobile apps. For this purpose 9185 apps have been manually and independently labeled for its gender distribution by 2 professional lexicographers. In this process, lexicographers have been instructed to label the gender distribution on the scale of 1-7. The meanings of these different label ids have been shown in Table 22. Descriptions of each app have been given as the source to judge its gender distribution. For example, the android games app “Blackjack Vegas”[25] would be played mostly by male users than the female users. Thus, it has been labeled as “1” by the professional lexicographers.

Table 22 - Gender Distribution Labels

Label ID	Meaning
1	80 % Male & 20% Female
2	70 % Male & 30% Female
3	60 % Male & 40% Female
4	50 % Male & 50% Female
5	40 % Male & 60% Female
6	30 % Male & 70% Female
7	20 % Male & 80% Female

[25] https://play.google.com/store/apps/details?id=com.mobilemediacom.blackjack

In order, to assess the reliability and validity of the rating, inter-judge raw agreement and Hit ratio were calculated. Inter-judge raw agreement was calculated by counting the number of items both judges labeled the same, divided by the total number of items (Moore & Benbasat, 1991). The hit ratio is the "overall frequency with which judges place items within is intended labels" (Moore & Benbasat, 1991). Results show that there are no major concerns with the labeling validity and reliability of these labels. Inter-rater raw agreement score, which averaged 0.89, exceeds the acceptable levels of 0.65 (Moore & Benbasat, 1991). The overall hit ratio of items was 0.90.

Once lexicographer finished labeling of all 9185 apps, the data set is divided into 2 sets for the training and testing purposes. For training and testing, 6170 and 3015 apps have been used respectively. Training instances are made containing fairly equal amounts of apps in each category (i.e. 1-7). For example, category 1 and category 2 are allocated with 337 and 391 apps respectively. This way it has been made sure the over fitting issues did not occur during the training process.

Further, by using the 9185 apps corpus is built with the respective tf-IDF score of each token. When building the corpus the, app descriptions and reviews were subjected to same preprocessing mechanism which was described in Step 2 of the App Classification sub section. Corpus is created by using the Apache Lucene Indexer[26]. Once the corpus is created, training model is built using the apps which

[26] http://lucene.apache.org/core/

have been identified for training purposes. We detail below the step by step procedure of this approach.

In this approach different feature selection methodologies such as Information Gain, Chi-Square, Top 15 Bi-grams and Unigrams and Top 10 Unigrams are used and its accuracy is evaluated. We have detailed below the steps taken using the Top 10 Unigram approach.

1. Each app description is fetched and subjected to preprocessing as discussed earlier (stemming, lemmatization and stop word removal).
2. For each app, top 10 descriptive tokens are identified using the relevant tf-IDF scores and then the master feature set is built. Altogether 20184 features have been identified for master feature set using the training data set apps.
3. Each app is then represented by these top 10 features. Respective feature's tf-IDF scores have been used as the numerical representational value.
4. Each app's gender label (1-7) has been used as the class variable and the rest of all the features have been used as the predictor variables.
5. Support Vector Machine Regression (Joachims, 1998) with Gaussian Radial Basis Function (RBF) kernel has been used to learn the patterns to predict the gender. For this purpose statistical tool R has been used as and the package "e1071" has been adopted as the relevant package. Then the training model is built.

6. Test data set apps also subjected to preprocessing and numerical vector transformation procedure as described for training data set apps.
7. Then the test data set apps have been fed into R with the trained model and relevant gender is predicted.

Now we discuss the experimental results for the proposed solutions in the following section.

4.3.3 Experimental Results

Having estimated the audience for each app using a hybrid methodology, the efficacy of the proposed solution is analyzed in three steps. First, it has been analyzed the accuracy of different classifiers used for predicting the relevant categories of an app. Secondly, accuracy of audience measurement using classification accuracy has been analyzed (“Age”, “Education” and “Has Children”). Finally, the efficacy of gender prediction is analyzed. Details of the experimental procedures are described below.

In order to measure the accuracy of different classification approaches used, a test data was built using 372 randomly chosen apps from popular categories such as Business, Entertainment, Education, Finance, Game and Style & Fashion. For the identified 372 apps, input dataset was built using the feature extraction procedure described above. Then it was subjected to different classification approaches as discussed above (Naïve Bayes, TF-IDF & SVM). Based on the classification, the top 5 predicted classes for each app were chosen and the classes were validated by the 3 professional lexicographers for appropriateness. In order to assess the reliability and validity of the rating, inter-judge raw agreement and Hit ratio were calculated. Inter-rater raw agreement score, which averaged 0.73, exceeds the acceptable levels of 0.65 (Moore & Benbasat, 1991). The overall hit ratio of items was 0.82.

Table 23 illustrates the accuracy of different classifiers across different categories. Overall TF-IDF achieved the highest accuracy of 78% compared to the other two classifiers. Thus, TF-IDF classifier has been chosen in estimating the audience.

After validating the accuracy of classifiers, it proceeds to evaluate accuracy of app audience estimation specifically the Age, Education and Has Children matrices. For this purpose, the same test data (i.e. 372 randomly chosen apps) that were used in measuring the accuracy of app classification. Following steps were carried out. First, professional lexicographers have been employed to manually estimate the audience of given apps using the relevant app store URL (e.g. https://itunes.apple.com/us/app/abc-sight-words-writing-free/id379874412?mt=8). Then the automated audience estimation process was carried out. Table 26 shows the estimated audience for the set of apps.

The efficacy of audience estimation was carried out by comparing the automated audience estimation (demographic) values with manually assigned demographic values using the well-known root-mean-square-error (MAPE) metric. For N (372 randomly chosen) apps, if $\mathrm{R}_1^0, \mathrm{R}_2^0, \dots, \mathrm{R}_\mathrm{N}^0$ is obtained as the estimated demographic values using the proposed approach and $\mathrm{R}_1^\mathrm{d}, \mathrm{R}_2^\mathrm{d}, \dots, \mathrm{R}_\mathrm{N}^\mathrm{d}$ as the manually assigned demographic values by professional lexicographers, then, the RMSE is computed as follows

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{\left(R_i^0 - R_i^d\right)}{R_i^0}$$

In this way, demographic dimensions such as "Age", "Education" and "Has Kids" achieved 85.5%, 80.9%, and 80.07% accuracies respectively.

Table 23 - Classification Accuracy across Categories

IAB Categories	***Accuracy***		
	Multinomial Naïve Bayes	TF-IDF	SVM
Business	62.50%	69.75%	71%
Style & Fashion	67.2%	79.6 %	68%
Arts & Entertainment	71.00%	83.00%	72%
News	68.14%	74.28%	67%
Health and Fitness	77.50%	93.75%	84.25%
Personal Finance	74.00%	80.00%	82%
Sports	74.00%	81.60%	78%
Education	68.00%	74.75%	67%
Hobbies & Interests	76.00%	73.00%	69.8%
Travel	74%	70%	76.5%
Overall:	71.234%	77.97%	73.56%

Table 24 - Parameter Values used in SVM Multiclass

Parameter name	Value
kernel	Gausssian RBF
cost	200
gamma	0.002
epsilon	0.1
# of support vectors	1

Above mentioned Table 24 shows the parameters which are used in classifying the apps in set of categories. These parameters were chosen since they were giving better accuracies. Further Gausian RBF is used since the relationship between class variables and tokens would be non-linear.

As the 3rd step efficacy of proposed gender prediction mechanism is evaluated. For this purpose 3015 apps and their respective descriptions are used as the source to build the test data. All the app descriptions were subjected to the same steps as discussed for training dataset apps (stemming, lemmatization and stop word removal). After this step different feature selection approaches such as Information Gain, Chi-Square, Top 15 bi-grams and 1-gram tokens and top 10 unigram tokens are used. Table 25 shows the number of features chosen while using different feature selection mechanism and their precision, recall and overall accuracy values. We observed that when using Top-10 unigrams higher accuracy is produced for predicting the gender of mobile applications. Thus, it has been identified that prediction accuracy increases when the matrix size is large. In this case it is 3015*20647.

Table 25 - Accuracy across different feature selection methods

Feature Selection Method	Total number of Features	Precision	Recall	Overall Accuracy
Information Gain	965	0.715	0.64	85%
Chi-Square	847	0.69	0.58	84.23%
Top-15 bi-grams & Unigrams	11370	0.81	0.72	87%
Top-10 Unigrams	20647	0.84	0.76	88.73%

Table 26 - Audience Estimation scores for apps

App Name	***Child 0-12***	***Teen 13-19***	***Gen Y20-34***	***Middle 35-64***	***Male***	***Female***	***College***	***Grad School***	***No College***	***Has Kids***	***Has No Kids***
ABC Sight Words Writing Free Lite HD - for iPad	0.95	0.05	0	0	0.45	0.55	0.01	0.06	0.92	0	1
Fantastic 4 In A Row Free	0.95	0.05	0	0	0.38	0.62	0	0	1	0	1
Chalkboard Addition	0.95	0.05	0	0	0.43	0.57	0	0.05	0.94	0	1
Doodle Hangman Free	0.1	0.9	0	0	0.41	0.59	0.02	0.04	0.94	0	1
Drexel Sports	0	0.9	0.1	0	0.35	0.65	0	0.06	0.94	0	1
Hello! (Ad-Free) texting and messaging	0	0.7	0.3	0	0.5	0.5	0.3	0	0.7	0	1
IM+	0	0.3	0.7	0	0.5	0.5	0.46	0.15	0.38	0.58	0.42
Beintoo	0	0.1	0.7	0.2	0.4	0.6	0.3	0.5	0.2	0.55	0.45
POF âˆ’ Free Online Dating for iPad	0	0.2	0.8	0	0.7	0.3	0.4	0.2	0.4	0.6	0.4
Formspring	0	0.1	0.8	0.1	0.5	0.5	0.44	0.16	0.39	0.65	0.35
Generation Next Youth	0	0.2	0.7	0.1	0.6	0.4	0.3	0.2	0.5	0.4	0.6
Today's Calendar with ads	0	0.01	0.24	0.75	0.55	0.45	0.64	0.24	0.11	0.92	0.08

4.3.4 Discussion & Conclusion

In this study, we have identified that important constituents of app ecosystem face numerous hurdles in estimating the right audience for mobile apps. In order to solve this problem, we have proposed a dynamic approach that can effectively measure the audience demographics for the millions of existing apps as well as the new incoming apps. Experimental results of the approach yield satisfactory results. This study has some important implications. Firstly, by using this audience estimation method both mobile advertisers and app developers can greatly benefit by precisely targeting consumers. Since most of the ad-requests do not contain relevant audience information, this approach can be used to plug this data as the third party platforms to the ad-requests. Secondly, the app platform owners (e.g. Apple and Android) can use both classification and audience measurement methods to effectively separate and estimate the audience for one thousand thousands of existing apps and incoming new apps, and therefore reach more consumers. Lastly, this audience estimation can also help mobile app users in distinguishing the most suitable app that can meet their demands and desires. Given the popularity and usefulness of mobile apps, studies of this nature can greatly help many constituents of app ecosystem and has a rich potential to extend the research of the e-business.

Chapter 5
Conclusion & Future Directions

Mobile apps are increasingly popular in various markets across the globe. The total number of apps in the mobile app market and their rate of growth are remarkable. An average user spends 10% of their media attention, staring at their smartphones and tablets. Statistics suggest that approximately 224 million people use mobile apps on a monthly base, compared to 221 million desktop users, i.e. mobile app users are somewhat more than desktop users. Moreover, we observe that mobile have become the first screen and made TV as the second screen during the recent super bowl event. This has resulted in the burgeoning mobile apps market, attracting many brand owners, who are keen on capitalizing business opportunities and targeting more customers through mobile advertising. Thus, mobile apps have become a lucrative medium with a growing customer base and promising revenue.

Thus motivated, we have focused on how programmatic media buying could help in designing effective mobile ad campaigns. In particular, we proposed that ad campaigns would be more effective when there is a way to determine the popularity signals in real time and when there is a way to pass the relevant audience information of mobile apps from which ad requests are coming. Further, we have elaborated on how the programmatic buying of social media popularity of an app, real time popularity rank computation of mobile apps and app audience information is vital in designing effective ad campaigns. The efficacies of proposed methodologies were subjected to rigorous experiment validation and the results were outperforming.

Particularly in the first study, we have addressed the problem of reliably identifying tweets related to mobile apps. Further, we addressed the aliasing and name conflict problems inherent in the task. The proposed approach has been compared with the Naïve Bayesian approach and a commercial implementation ("Socialmention"). The proposed approach outperformed in all measures of accuracy compared to Bayesian approach and the "Socialmention". While the proposed approach has been validated in mobile app domain the techniques are generally applicable to other domains as well.

In the second study, we have addressed the problem of computing popularity ranks of mobile apps. In particular, a novel manner of weight calculation has been introduced based on alteration in signal strengths of the features and re-ordering weights based on current relative signal intensities. The approach, termed DOWA, was validated for accuracy against established ground truth values as well as against state-of-the-art OWA methods. The results were very encouraging: in an absolute sense, DOWA was consistently within 10% of ground truth values, exhibiting high absolute accuracy of DOWA. We strongly believe that, this study also makes several significant contributions to important constituents of app ecosystem. First, the true app rank can help consumers and advertisers who are constantly faced with the dilemma of choosing. It would be vital for advertisers knowing the app popularity ranks in real time and programmatic buying of this information would increase the effectiveness of mobile advertising. The practical impact of DOWA goes beyond app rank computation. It represents a robust technique which can be used in any scenario possessing similar characteristics, namely dynamically varying signal strengths.

In the third study, we have addressed the problem of estimating the right audience for mobile apps. In order to solve this problem, a dynamic approach which can effectively measure the audience demographics for the millions of existing apps as well as the new incoming apps is proposed. Experimental results of the approach yield satisfactory results. Since most of the ad-requests do not contain relevant audience information, this approach can be used to plug this data as the third party platforms to the ad-requests. Given the popularity and usefulness of mobile apps, studies of this nature can greatly help many constituents of app ecosystem.

As a next step, the effectiveness of incorporating proposed programmatic buying framework on real time mobile advertisement campaigns would be evaluated. For example, suppose an advertiser wants to design an ad campaign for a newly designed female fashion outfit. For this purpose, the proposed framework can be utilized. Initially, using audience profiling of each app (i.e. study 3), advertisers can identify the apps which are mostly used by female users. Then the most popular female apps at a given time *t* can be identified by deriving the overall popularity of an app in time t (using study 2). In addition, popular fashion apps into Social media can also be considered (i.e. using study 1). Subsequently, the ads campaign can be delivered through the set of identified apps.

In order to measure the effectiveness of this programmatic buying framework in an ad campaign, several experiments will be designed. First, an experiment will be designed for ads only using app audience profiles, then a second experiment would be designed for ad campaign using only popularity signals and a third experiment would be designed for ad campaign targeting the apps which are popular at time *t* and being used mostly by female users. Finally, an experiment that combines all three approaches would be designed. The practical efficacy of the programmatic buying framework will be validated against using direct ad serving frameworks (i.e. non-programmatic buying) and programmatic buying without providing the additional info such as app audience profile information and app popularity signals (including social media). This way the effectiveness of each and combined approaches can be examined in an ad-campaign at a given time.

References

Aalto, L., Gothlin, N., Korhonen, J., and Ojala, T. 2004. “Bluetooth and WAP Push Based Location-Aware Mobile Advertising System,” *Proceedings of the 2Nd International Conference on Mobile Systems, Applications, and Services* pp. 49–58 ACM

Adar, E. A. R. C. 2007. “User Profile Classification by Web Usage Analysis,” U.S.A: U S C

Alchemyapi.com. 2012. “AlchemyAPI: Transfroming Text Into Knowledge,.” Retrieved from http://www.alchemyapi.com/products/features/sentiment-analysis/

Alexa. 2012. “Alexa the Web Information Company,.” Retrieved from http://www.alexa.com

Amblee, N., and Bui, T. X. 2007. “Freeware Downloads: An Empirical Investigation Into the Impact of Expert and User Reviews On Demand for Digital Goods,” In J. A. Hoxmeier & S. Hayne (Eds.) *Proceedings of America’s Conference Information System* p. 21 Association for Information Systems

Amin, G. R., and Emrouznejad, A. 2006. “An extended minimax disparity to determine the OWA operator weights,” *Computers and Industrial Engineering* 50(3), pp.312–316

Androutsopoulos, I., Koutsias, J., Chandrinos, K. V, Ch, K. V, Paliouras, G., and Spyropoulos, C. D. 2000. “An Evaluation of Naive Bayesian Anti-Spam Filtering,” *Proceedings of the workshop on Machine Learning in the New Information Age* pp. 9–17

Apache. 2012. “Open NLP,.” Retrieved from https://opennlp.apache.org/

App Brain. 2012, December. “Apps by Downloads,.” Retrieved from http://www.appbrain.com/stats/android-app-downloads

Asur, S., and Huberman, B. A. 2010. “Predicting the Future with Social Media,” *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology* WI-IAT ’10 pp. 492–499 IEEE Computer Society

Bagwell, K. 2007. “*The Economic Analysis of Advertising*,” (M. Armstrong & R. Porter, Eds.) 3rd ed.

Banerjee, S. 2007. “Clustering Short Texts using Wikipedia,” *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* ACM

Bertrand, S. 2013. “App Annie,.” Retrieved from http://www.crunchbase.com/company/app-annie

Bollen, J., Mao, H., and Pepe, A. 2011. “Modeling public mood and emotion: Twitter sentiment and socioeconomic phenomena,” *Proceedings of the 5th International Conference on Weblogs and Social Media* pp. 450–453

Borgs, C., Chayes, J., Borgs, C., Etesami, O., Immorlica, N., and Mahdian, M. 2007. “Dynamics of bid optimization in online advertisement auctions,” *Proceedings of the 16th International World Wide Web Conference* pp. 531–540

De Bruyn, A., and Lilien, G. L. 2008. “A multi-stage model of word-of-mouth influence through viral marketing,” *International Journal of Research in Marketing* 25(3), pp.151–163

Carlsson, C., and Fuller, R. 1997. “OWA operators for decision support,” *Proceedings of EUFIT97 Conference* pp. 1539–1544

Cary, M., Das, A., Edelman, B., Giotis, I., Heimerl, K., Karlin, A., Mathieu, C., and Schwarz, M. 2007. “Greedy bidding strategies for keyword auctions,” *Proceedings of 9th ACM Conference on Electronic Commerce* pp. 262–271 ACM

Chang, C. C., and Lin, C. J. 2001. “LIBSVM: a library for support vector machines,.” Retrieved from http://www.csie.ntu.edu.tw/~cjlin/libsvm

Chevalier, J. A., and Mayzlin, D. 2003. “*The effect of word of mouth on sales: Online book reviews*” National Bureau of Economic Research

Chevalier, J. A., and Mayzlin, D. 2006. “The Effect of Word of Mouth on Sales: Online Book Reviews,” *Journal of Marketing Research* 43(3), pp.345–354 National Bureau of Economic Research Cambridge, Mass., USA

Chintagunta, P., and Vilcassim, N. 1992. “An Empirical Investigation of Advertising Strategies in a Dynamic Duopoly,” *Management Science* 38(9), pp.1230–1244

Cho, C. H., and Khang, H. 2006. “The state of internet-related research in communications, marketing and advertising:1994–2003,” *Journal of Advertising* 35(3), pp.143–163

Chu, S.-C., and Kim, Y. 2011. “Determinants of consumer engagement in electronic word-of-mouth (eWOM) in social networking sites,” *International Journal of Advertising* 30(1), pp.47–75

Cohen, K. 2013. “Driving mobile app success with word of mouth marketing,.” Retrieved from https://www.womma.org/blog/2013/02/driving-mobile-app-success-with-word-of-mouth-marketing

Crofford, A. 2011, November. “7 Ways to Have a Top Grossing App ? How 4 Apps did it,.” Retrieved from http://mobileorchard.com/7-ways-to-have-a-top-grossing-app-how-4-apps-did-it/

Cui, H., Wen, J.-R., Nie, J.-Y., and Ma, W.-Y. 2003. “Query expansion by mining user logs,” *IEEE Transactions on Knowledge and Data Engineering* 15(4), pp.829–839

Danaher, P., and Mullarkey, G. 2003. “Factors Affecting Online Advertising Recall: A Study of Students,” *Journal of Advertising Research* 44(3), pp.252–257

Das, S. R., and Chen, M. Y. 2007. “Yahoo! for Amazon: Sentiment Extraction from Small Talk on the Web,” *Management Science* 53(9), pp.1375–1388 Institute for Operations Research and the Management Sciences (INFORMS), Linthicum, Maryland, USA: INFORMS

Datta, A. 2013. “Why Is the Success of Mobile Apps So Difficult to Measure? Critical Issue in Audience Measurement Unraveled,” *Huffingtonpost* . Retrieved from http://www.huffingtonpost.com/anindya-datta/success-of-mobile-apps_b_2860915.html

Davenport, T. H., and Beck, J. C. 2001. “The attention economy: Understanding the new currency of business,” *Cambridge, MA: Harvard Business Press*

Dellarocas, C., Zhang, X. M., and Awad, N. F. 2007. “Exploring the value of online product reviews in forecasting sales: The case of motion pictures,” *Journal of Interactive Marketing* 21(4), pp.23–45

Dent, K., and Paul, S. A. 2011. “Through the Twitter Glass: Detecting Questions in Micro-text,” *Proceedings of AAAI-11 Workshop on Analyzing Microtext*

Duan, W., Gu, B., and Whinston, A. 2008. “The dynamics of online word-of-mouth and product sales An empirical investigation of the movie industry,” *Journal of Retailing* 84(2), pp.233–242 Elsevier

Ebbert, J. 2012. “Define It - What Is Programmatic Buying?,.” Retrieved from http://www.adexchanger.com/online-advertising/define-programmatic-buying/

Edelman, and Schwarz, M. 2011. “*Optimal Auction Design and Equilibrium Selection in Sponsored Search Auctions* (No. 10-054)”

Erickson, G. 1992. “Empirical Analysis of Closed-Loop Duopoly Advertising Strategies,” *Management Science* 38(12), pp.1732–1749

Fausett, L. V. 1994. “*Fundamentals of neural networks: architectures, algorithms, and applications*,” Prentice-Hall, Inc.

Feng, Q., Hwang, K., and Dai, Y. 2009. “Rainbow Product Ranking for Upgrading E-Commerce.,” *IEEE Internet Computing* 13(5), pp.72–80

Filtertweets. 2012. “Filter Tweets for Greasemonkey,”

Fisher, S. R. A., Genetiker, S., Fisher, R. A., Genetician, S., Britain, G., and Généticien, S. 1970. “*Statistical methods for research workers*,” Vol. 14 Oliver and Boyd Edinburgh

Fuller, R., and Majlender, P. 2003. “On obtaining minimal variability OWA operator weights,” *Fuzzy Sets and Systems* 136(2), pp.203–215

Ghose, A., and Ipeirotis, P. G. 2007. “Designing novel review ranking systems: Predicting the usefulness and impact of reviews,” *Proceedings of the Ninth International conference on Electronic commerce ICEC07* pp. 303–310 ACM

Ghosh, A., Rubinstein, B.I.P. Vassilvitskii, S., and Zinkevich, M. 2009. "Adaptive bidding for display advertising," *Proceedings of the 18th International World Wide Web Conference* pp. 251–260

Girardello, A., and Michahelles, F. 2010. "AppAware: which mobile applications are hot?," *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services* MobileHCI '10 pp. 431–434 ACM

Goldfarb, A., and Tucker, C. 2011. "Privacy Regulation and Online Advertising," *Management Science* 57(1), pp.57–71

Google Inc. 2011. "Google Search Appliance Help Center,." Retrieved from https://support.google.com/gsa/?hl=en#topic=2707703

Hennig-Thurau, T., Gwinner, K. P., Walsh, G., and Gremler, D. D. 2004. "Electronic word-of-mouth via consumer-opinion platforms: what motivates consumers to articulate themselves on the internet?," *Journal of interactive marketing* 18(1), pp.38–52

Hevner, A. R., March, S. T., Park, J., and Ram, S. 2004. "Design Science in Information Systems Research," (A. R. Hevner & S. Chatterjee, Eds.)*MIS Quarterly* Integrated Series in Information Systems28(1), pp.75–105 JSTOR

Hoogsteder, V. 2013. "Distimo,." Retrieved from http://www.distimo.com/

Hu, J., Zeng, H., Li, H., Niu, C., and Chen, Z. 2007. "Demographic prediction based on user's browsing behavior," *Proceedings of the 16th international conference on World Wide Web* pp. 151–160

Hu, X., Sun, N., Zhang, C., and Chua, T.-S. 2009. "Exploiting internal and external semantics for the clustering of short texts using world knowledge," *CIKM '09: Proceeding of the 18th ACM conference on Information and knowledge management* pp. 919–928 ACM

IAB. 2011. "*Networks & Exchanges Quality Assurance Guidelines*." Retrieved from http://www.iab.net/media/file/IAB-NE-QA-Guidelines-v1.5-November-2011-FINAL.pdf

Intridea. 2011. "TweetSentiment,." Retrieved from http://www.intridea.com/tweetsentiments

Jansen, B. J., Liu, Z., Weaver, C., Campbell, G., and Gregg, M. 2011. “Real time search on the web: Queries, topics, and economic value,” *Information Processing & Management* 47(4), pp.491–506 Tarrytown, NY, USA: Pergamon Press, Inc.

Joachims, T. 1998. “Text Categorization with Support Vector Machines: Learning with Many Relevant Features,” *ECML ’98 Proceedings of the 10th European Conference on Machine Learning* pp. 137–142 Springer-Verlag London, UK

Joanna, O., and Greene, M. 2012. “*The Future of Digital Media Buying*.” Retrieved from http://www.forrester.com/The+Future+Of+Digital+Media+Buying/fulltext/-/E-RES58354

Johnson, J., and Myatt, D. 2006. “On the Simple Economics of Advertising, Marketing, and Product Design,” *The American Economic Review* 96(3), pp.756–784

Kabbur, S., Han, E., and Karypis, G. 2010. “Content-based methods for predicting web-site demographic attributes,” *Proceedings of IEEE 10th International Conference on Data Mining (ICDM)* pp. 863–868

Kats, R. 2012. “App discovery still an ongoing struggle for marketers,.” Retrieved from http://www.mobilemarketer.com/cms/news/content/13731.html

Kwak, H., Lee, C., Park, H., and Moon, S. 2010. “What is Twitter, a Social Network or a News Media?,” *Proceedings of the 19th International Conference on World Wide Web* WWW ’10 pp. 591–600 ACM

Lee, M., Rodgers, S., and Kim, M. 2009. “Effects of valence and extremity of eWOM on attitude toward the brand and website,” *Journal of Current Issues & Research in Advertising* 31(2), pp.1–11

Leppäniemi, M., and Karjaluoto, H. 2005. “Factors Influencing Consumers’ Willingness to Accept Mobile Advertising. A Conceptual Model,” *International Journal of Mobile Communications* 3(3), pp.197–213

Lewis, D. D. 1998. “Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval,” pp. 4–15 Springer Verlag

Li, H., Bhowmick, S. S., and Sun, A. 2010. “Affinity-driven prediction and ranking of products in online product review sites,” CIKM ’10 pp. 1745–1748 ACM

MacQueen, J. B. 1967. “Some Methods for Classification and Analysis of MultiVariate Observations,” In L. M. Le Cam & J. Neyman (Eds.) *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability* Vol. 1, pp. 281–297 University of California Press

Mary, E. G. 2013. “There’s An App Audience for That, But It’s Fragmented,.” Retrieved from http://blog.flurry.com/bid/96368/There-s-An-App-Audience-for-That-But-It-s-Fragmented

Massad, V., and Tucker, J. 2000. “Comparing bidding and pricing between in-person and online auctions,” *Journal of Product and Brand Management* 9(5), pp.325–332

Mathioudakis, M., and Koudas, N. 2010. “TwitterMonitor: Trend detection over the Twitter stream.,” *Proceedings of the International Conference on Management of Data* pp. 1155–1158

Mayer-Schönberger, V. 1998. “The Internet and Privacy Legislation: Cookies for a Treat?,” *Computer Law & Security Review* 14(3), pp.166–174 Elsevier Advanced Technology

MicrosoftCorporation. 2011. “Bing Translator,.” Retrieved from http://www.bing.com/translator

Mobilewalla. 2012. “Mobilewalla-an app search engine,.” Retrieved from www.mobilewalla.com

Mobilewalla. 2013. “Mobilewalla Analytics,.” Retrieved from http://analytics.mobilewalla.com

Mohanty, B. K., and Passi, K. 2006. “Web based information for product ranking in e-business: a fuzzy approach,” ICEC ’06 pp. 558–563 ACM

Murray, D., and Durrell, K. 2000. “Inferring demographic attributes of anonymous internet users,” *Web Usage Analysis and User Profiling* pp. 7–20 Springer

Muthukrishnan, S. 2008. “Internet Ad Auctions: Insights and Directions,” *Proceedings of the 35th International Colloquium on Automata, Languages and Programming* pp. 14–23 Reykjavik, Iceland

Muthukrishnan, S. 2009. “Ad exchanges: research issues,” *Springer Lecture Notes in Computer Science (5929)* pp. 1–12 Springer

Nigam, K. 1999. “Using maximum entropy for text classification,” *In IJCAI-99 Workshop on Machine Learning for Information Filtering* pp. 61–67

O’Hagan, M. 1988. “Aggregating template or rule antecedents in real-time expert systems with fuzzy set Logic,” *Signals, Systems and Computers, 1988. Twenty-Second Asilomar Conference on* Vol. 2, pp. 681–689

Oghina, A., Breuss, M., Tsagkias, M., and Rijke, M. 2012. “Predicting IMDB Movie Ratings Using Social Media,” In R. Baeza-Yates, A. Vries, H. Zaragoza, B. B. Cambazoglu, V. Murdock, R. Lempel, & F. Silvestri (Eds.) *Advances in Information Retrieval SE - 51* Lecture Notes in Computer Science Vol. 7224, pp. 503–507 Springer Berlin Heidelberg

One Riot. 2009. “The inner workings of a realtime search engine,.” Retrieved from http://53tech.com/blog/snippets/274-oneriot-inner-workings-of-a-realtime-search-engine

Opera. 2012. “The State of Mobile Advertising, Q2 2012,” *Opera Mediaworks* . Retrieved from http://business.opera.com/sma/2012/q2/

PRNewswire. 2011. “MTV networks’ mobile apps study reveals the life cycle of an app: from discovered to discarded,.” Retrieved from http://www.prnewswire.com/news-releases/mtv-networks-mobile-apps-study-reveals-the-life-cycle-of-an-app-from-discovered-to-discarded-123348433.html

Quantcast. 2013. “May Mobile OS Share North America,” Quantcast. Retrieved from https://www.quantcast.com

Rappaport, T. 2001. “*Wireless Communications: Principles and Practice*,” 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR

Reinstein, D. A., and Snyder, C. M. 2005. “The influence of expert reviews on consumer demand for experience goods: A case study of movie critics,” *Journal of Industrial Economics* 53(1), pp.27–51

Richins, M. L., and Root-Shaffer, T. 1988. “The role of evolvement and opinion leadership in consumer word-of-mouth: An implicit model made explicit,” *Advances in consumer research* 15(1), pp.32–36

Rocchio, J. 1971. “*Relevance Feedback in Information Retrieval. In The SMART Retrieval System - Experiments in Automatic Document Processing*,” Prentice-Hall

Rui, H., and Whinston, A. 2011. “Designing a social-broadcasting-based business intelligence system.,” *ACM Transactions on Management Information Systems (TMIS)*, 2(4), pp.22

Ryan Kim. 2011. “Appsfire scores 3.6m as app discovery demands grow,” Gigaom. Retrieved from http://gigaom.com/2011/05/30/appsfire-scores-3-6m-as-app-discovery-demands-grow/

Sahami, M., Dumais, S., Heckerman, D., and Horvitz, E. 1998. “A Bayesian Approach to Filtering Junk E-Mail,” *Learning for Text Categorization: Papers from the 1998 Workshop* Madison, Wisconsin: AAAI Technical Report WS-98-05

Salton, G., and McGill, M. J. 1986. “*Introduction to Modern Information Retrieval*,” New York, NY, USA: McGraw-Hill, Inc.

Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., and Sperling, J. 2009. “TwitterStand: news in tweets,” *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems* GIS ’09 pp. 42–51 ACM

Sarah, P. 2013. “Apple’s App Store Rankings Algorithm Changed To Consider Ratings, And Possibly Engagement,” *Techcrunch* . Retrieved from http://techcrunch.com/2013/08/23/apples-app-store-rankings-algorithm-changed-to-favor-ratings-and-possibly-engagement/

Sarkas, N., Bansal, N., Das, G., and Koudas, N. 2009. “Measure-driven Keyword-Query Expansion,” *Proceedings of The Vldb Endowment* 2, pp.121–132

Schneider, K.-M. 2003. “A Comparison of Event Models for Naive Bayes Anti-Spam E-Mail Filtering,” *In Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL-03* pp. 307–314

Shannon, C. E. 1948. “A Mathematical Theory of Communication,” *The Bell System Technical Journal* 27, pp.379–423

Simon, K. 2013. “The Rise of the App & Mortar Economy,” Flurry. Retrieved from http://www.flurry.com/bid/93560/The-Rise-of-the-App-Mortar-Economy

SocialMention. 2012. “SocialMention,.” Retrieved from http://socialmention.com

Spärck Jones, K. 1972. "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation* 28(1), pp.pp. 11 – 21

Sriram, B., Fuhry, D., Demir, E., Ferhatosmanoglu, H., and Demirbas, M. 2010. "Short text classification in twitter to improve information filtering," *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval* SIGIR '10 pp. 841–842 ACM

Szabo, G., and Huberman, B. A. 2010. "Predicting the popularity of online content," *Communications of ACM* 53(8), pp.80–88 New York, NY, USA: ACM

Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., and Kappas, A. 2010. "Sentiment strength detection in short informal text," *Journal of the American Society for Information Science and Technology* 61(12), pp.2544–2558

Toutanova, K., Klein, D., Manning, C. D., and Singer, Y. 2003. "Feature-rich part-of-speech tagging with a cyclic dependency network," *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1* NAACL '03 pp. 173–180 Association for Computational Linguistics

Trusov, M., Bucklin, R. E., and Pauwels, K. 2009. "Effects of word-of-mouth versus traditional marketing: Findings from an internet social networking site," *Journal of Marketing* 73, pp.90–102

Tsang, M.L., Ho, S.C. and Liang, T. P. 2004. "Consumer attitudes toward mobile advertising: an empirical study," *International Journal of Electronic Commerce* 8(3), pp.65–78

TweetFilter. 2012. "Tweetfilter for Greasemonkey,"

Varshney, U., and Vetter, R. 2002. "Mobile commerce: Framework, applications, and networking support," *Journal of Mobile Networks and Applications* 7(3), pp.185–193

Vega, T. 2012. "The New Algorithm of Web Marketing,." Retrieved from http://www.nytimes.com/2012/11/16/business/media/automated-bidding-systems-test-old-ways-of-selling-ads.html

ViralHeat. 2012. "ViralHeat,." Retrieved from https://www.viralheat.com/solutions/sentiment-analysis/

Vollmer, C., and Precourt, G. 2008. "*Always on: advertising, marketing, and media in an era of consumer contro*," New York: McGraw-Hill Education.

Xu, Z. 2006. "Dependent OWA operators," *Proceedings of the Third international conference on Modeling Decisions for Artificial Intelligence* MDAI'06 pp. 172–178 Berlin, Heidelberg: Springer-Verlag

Yager, R. R. 1988. "On ordered weighted averaging aggregation operators in multicriteria decisionmaking," *IEEE Transactions on Systems, Man and Cybernetics* 18(1), pp.183–190

Yager, R. R. 1993. "Families of OWA operators," *Fuzzy Sets and Systems* 59(2), pp.125–148 Elsevier

Yager, R. R., and Filev, D. P. 1999. "Induced ordered weighted averaging operators.," *IEEE transactions on systems man and cybernetics Part B Cybernetics a publication of the IEEE Systems Man and Cybernetics Society* 29(2), pp.141–150

Yan, J., Liu, N., Wang, G., Zhang, W., Jiang, Y., and Chen, Z. 2009. "How much can behavioral targeting help online advertising?," *Proceedings of the 18th International World Wide Web Conference* pp. 261–270

Yin, P., Luo, P., Wang, M., and Lee, W.-C. 2012. "A straw shows which way the wind blows: ranking potentially popular items from early votes," WSDM '12 pp. 623–632 ACM

Yu, B., Chen, M., and Kwok, L. 2011. "Toward predicting popularity of social marketing messages," SBP'11 pp. 317–324 Springer-Verlag

Zhang, B., Dai, H., Zeng, H.-J., Qi, L., Najm, T., Mah, T. B., Shipunov, V., et al. 2006. "Predicting demographic attributes based on online behavior," Google Patents. Retrieved from https://www.google.com/patents/US20070208728

Zhang, L., Zhu, J., and Yao, T. 2004. "An evaluation of statistical spam filtering techniques," *ACM Transactions on Asian Language Information Processing (TALIP* 3, pp.2004

Zhang, Z., Li, X., and Chen, Y. 2012. "Deciphering word-of-mouth in social media: text-based metrics of consumer reviews.," *ACM Transactions on Management Information Systems (TMIS)* 3(1), pp.5:1–5:23