Smartphone App Categorization for Interest Targeting in Advertising Marketplace

Vladan Radosavljevic, Mihajlo Grbovic,
Nemanja Djuric, Narayan Bhamidipati
Yahoo Labs, Sunnyvale, CA, USA
{vladan, mihajlo, nemanja,
narayanb}@yahoo-inc.com

Daneo Zhang, Jack Wang,
Jiankai Dang, Haiying Huang,
Ananth Nagarajan, Peiji Chen
Yahoo Inc., Sunnyvale, CA, USA
{daneozxy, jackwg, dangj, hhuang, ananth,
peiji}@yahoo-inc.com

ABSTRACT

Last decade has witnessed a tremendous expansion of mobile devices, which brought an unprecedented opportunity to reach a large number of mobile users at any point in time. This resulted in a surge of interest of mobile operators and ad publishers to understand usage patterns of mobile apps and allow more relevant content recommendations. Due to a large input space, a critical step in understanding app usage patterns is reducing sparseness by classifying apps into predefined interest taxonomies. However, besides short name and noisy description majority of apps have very limited information available, which makes classification a challenging task. We address this issue and present a novel method to classify apps into interest categories by: 1) embedding apps into low-dimensional space using a neural language model applied on smartphone logs; and 2) applying k-nearest-neighbors classification in the embedding space. To validate the method we run experiments on more than one billion device logs covering hundreds of thousands of apps. To the best of our knowledge this is the first app categorization study at this scale. Empirical results show that the proposed method outperforms the current state-of-the-art.

1. INTRODUCTION

Global count of mobile devices increased to 7.4 billion by the end of 2014[1] with almost half a billion additional mobile devices in 2014 alone, thus surpassing the world population. In addition, global mobile data traffic grew by almost 70% in 2014. The expansion of number and usage of mobile devices is closely followed by an enormous number of mobile apps developed for these devices. Apps are mainly available through dedicated stores for each operating system (e.g., Google Play Store for Android apps, Apple App Store for iOS, or Windows Phone Store for Windows), where there are more than 3.5 million available apps as of July 2015[2].

Mobile devices and apps in particular play an increasingly important role in our daily habits. Users use them on a daily basis to check weather, communicate with friends, read news, or play video games. Better understanding of these usage patterns of mobile apps can help in inferring user preferences, which can be used by mobile operators, publishers, and app developers to improve personalized services such as content recommendation and have more effective monetization through personalized interest advertising. However, the app space is huge, and analysis of usage patterns at an app level easily becomes an unfeasible approach. Thus, a critical, often taken step is to reduce sparseness of the input space by classifying apps into predefined interest categories.

The apps in stores are labeled according to store-specific, high-level classification schemes. However, such classification schemes are often too coarse to be suitable for user modeling. For example, app "10 Daily Exercises" is classified under category "Health & Fitness", the same category where app "BabyBump Pregnancy" is classified into. In addition, it is up to the developers themselves to label the apps, which introduces a lot of noise. Recently, a method for automated app classification was proposed that relies on expanded app meta data by issuing app name as a query to web search engine, and collecting search result snippets [4]. The method is evaluated on a small number of apps, while the scalability to large number of apps is an open question. In [1] app metadata in a form of app description available at the app stores was used to construct app features which are then fed into supervised classifier. The limitation of this approach is the fact that it relies solely on the noisy textual description available from the app stores.

In this paper we present a novel method for app classification into a fine-grained interest taxonomy. Motivated by recent success of language models in a number of natural language processing tasks [3], we propose to use a language model to learn app embeddings in a low-dimensional space, in which similar apps reside nearby. Following the embedding step, we propose to use a k-nearest neighbor classification approach to classify apps into interest categories.

2. METHODOLOGY

Let us assume we are given a set of apps $\mathcal{A} = \{a_j | j = 1 \dots M\}$, each identified by a unique identifier a_j. Each app is associated with app meta data that includes app name and textual description from app stores. In addition, app install times extracted from app usage logs of N users over a time period T are also known. For the i^{th} user we collect data

[1] http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html, accessed February 2016

[2] http://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/, accessed Feb. 2016

WWW'16 Companion, April 11–15, 2016, Montréal, Québec, Canada.
ACM 978-1-4503-4144-8/16/04.
http://dx.doi.org/10.1145/2872518.2889411.

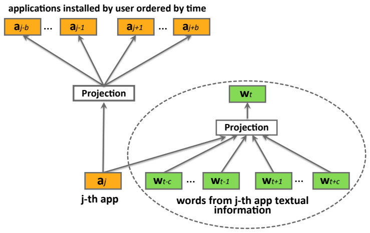

Figure 1: Graphical representation of app2vec model

Table 1: Qualitative analysis of learned app embeddings (nearest neighbors of app "Daily Workouts Free")

Nearest neighbor	Cosine similarity
Daily Cardio Workout Free	0.935
Daily Leg Workout Free	0.934
Daily Ab Workout Free	0.932
Daily Butt Workout Free	0.927
7-minute Workout	0.899

in a form $p_i = \{(a_j, t_j), j = 1, \ldots, K_i, t_1 < t_2 < \ldots < t_{K_i}\}$, where p_i denotes the user's profile, K_i is number of apps that the user installed, and t_j is install time of app a_j.

We consider the task of app classification, where we classify apps into one or more categories of a predefined interest taxonomy. In our work we leverage an in-house taxonomy, amounting to around 250 categories. The categories cover a variety of fine-grained interest topics, such as "Hobbies and activities/Pets/Cats" or "Entertainment/Movies/Action".

2.1 Proposed approach

In the context of natural language processing, distributed models learn word vectors in a low-dimensional space using context of a word within a sentence, such that semantically related words are close in the embedding space [3]. Recently, distributed models were extended to a two-level architecture capable of learning low-dimensional representations of documents by leveraging document stream as document context and document words as document content [2]. Motivated by [2], we propose app2vec (shown in Figure 1), a two-level architecture where the upper layer models temporal context of app install sequences using the skip-gram [3], while the bottom layer models word sequences within app metadata using continuous bag-of-words [3].

In particular, given a set $\mathcal{P}$ of N user profiles, w.l.o.g profile $p \in \mathcal{P}$ is defined as an install sequence of K apps and each app is described by name and description that consist of L words, $a = (w_1, \ldots, w_L)$. The objective is to maximize log-likelihood of the training data,

$$\begin{aligned}\mathcal{L} = &\frac{1}{N} \sum_{p \in \mathcal{P}} \Big(\sum_{a_j \in p} \sum_{-b \leq i \leq b, i \neq 0} \log \mathbb{P}(a_{j+i} | a_j) \\ &+ \sum_{a_j \in p} \alpha_j \sum_{w_t \in a_j} \log \mathbb{P}(w_t | w_{t-c} : w_{t+c}, a_j) \Big),\end{aligned} \tag{1}$$

where weights α trade off between minimization of the log-likelihood of app install sequences (or app *context*) and word sequences (or app *content*), while b and c are context widths for app install sequences and app descriptions, respectively. Denoting install frequency of the j^{th} app as F_j, we set $\alpha_j = \frac{1}{\log(1+F_j)}$, such that rarely installed apps rely more on content, whereas frequently installed apps rely more on context. Probability $\mathbb{P}(a_{j+i}|a_j)$ of observing a neighboring app given the current app is defined using softmax, while probability of observing a content word $\mathbb{P}(w_t|w_{t-c} : w_{t+c}, a_j)$ depends not only on its surrounding words, but also on the app that the word belongs to [2].

Table 2: Relative improvements in precision and recall of the competing methods with respect to the baseline

Algorithm	Precision	Recall
tf-idf	-	-
LDA	-12%	-0.1%
context2vec	+20%	+2.8%
context-content2vec	**+34%**	**+2.8%**

To classify an unlabeled app a_u we use a small editorially labeled set of apps and perform the following steps: 1) look up vector representation of a_u; 2) find nearest labeled apps based on cosine similarity in the embedding space; 3) apply voting mechanism among k-nearest neighbors, where all labels with at least x% of votes are assigned to a_u.

3. EXPERIMENTS

We learned app embeddings using large-scale proprietary mobile logs. Number of apps in app2vec vocabulary was subsampled to 200,000. The window lengths were set to $b = 5$ and $c = 10$. We obtained editorially labeled set of 10,000 apps, uniformly distributed over 250 interest labels. For classification of unlabeled apps we used $k = 10$ neighbors, where as neighbors we considered all labeled apps with cosine similarity greater than 0.8. We determined voting threshold $x = 30\%$ through cross-validation.

In Table 1 we show nearest neighbors of app "Daily Workouts Free", where we can see that semantically similar apps are nearby in the embedding space. To quantify accuracy of proposed approach, we compared our classification strategy to logistic regression model trained of features constructed from: 1) tf-idf representation of app description text; and 2) LDA applied on app description text. We also compared the proposed two-level architecture (*context-content2vec*) to learning representations based on app install sequences only (*context2vec*). The relative improvements of precision and recall metrics with respect to the baseline tf-idf are reported in Table 2, where we can see that the proposed method achieves superior performance over the alternatives.

4. REFERENCES

[1] G. Berardi, A. Esuli, T. Fagni, and F. Sebastiani. Multi-store metadata-based supervised mobile app classification. In *Proceedings of the 30th Annual ACM Symposium on Applied Computing, Salamanca, Spain, April 13-17, 2015*, pages 585–588, 2015.

[2] N. Djuric, H. Wu, V. Radosavljevic, M. Grbovic, and N. Bhamidipati. Hierarchical neural language models for joint representation of streaming documents and their content. In *International World Wide Web Conference (WWW)*, 2015.

[3] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[4] H. Zhu, E. Chen, H. Xiong, H. Cao, and J. Tian. Mobile app classification with enriched contextual information. *IEEE Trans. Mob. Comput.*, 13(7):1550–1563, 2014.