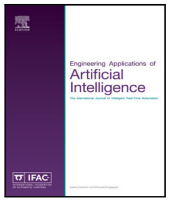
Engineering Applications of Artificial Intelligence 121 (2023) 105974

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Semantic similarity for mobile application recommendation under scarce user data

João Coelho [a,b,*], Diogo Mano [a,c], Beatriz Paula [a,b], Carlos Coutinho [a,d,e], João Oliveira [d,e,f], Ricardo Ribeiro [d,g], Fernando Batista [d,g]

[a] Caixa Mágica Software, Lisboa, Portugal
[b] Instituto Superior Técnico, Lisboa, Portugal
[c] Faculdade de Ciências, Lisboa, Portugal
[d] Instituto Universitário de Lisboa (ISCTE-IUL), Lisboa, Portugal
[e] Centro de Investigacão em Ciências da Informacão, Tecnologias e Arquitetura (ISTAR-IUL), Lisboa, Portugal
[f] Instituto de Telecomunicacões, Lisboa, Portugal
[g] INESC-ID Lisboa, Lisboa, Portugal

ARTICLE INFO

Keywords:
Recommendation systems
More like this recommendation
Semantic similarity
Mobile applications
Transformers

ABSTRACT

The *More Like This* recommendation approach is ubiquitous in multiple domains and consists in recommending items similar to the one currently selected by the user, being particularly relevant when user data is scarce. We studied the impact of using semantic similarity in the context of the *More Like This* recommendation for mobile applications, by leveraging dense representations in order to infer the similarity between applications, based on their textual fields. Our approach was validated by comparing it to the solution currently in use by Aptoide, a mobile application store, since no benchmarks are available for this specific task. To further evaluate the proposed model, we asked 1262 users to compare the results achieved by both approaches, also allowing us to build an annotated dataset of similar applications. Results show that the semantic representations are able to capture the context of the applications, with more useful recommendations being presented to users, when compared to Aptoide's current solution. For replication and future research, all the code and data used in this study was made publicly available, including two novel datasets (installed applications for more than one million users, and app user-labeled similarity), the fine-tuned model, and the test platform.

1. Introduction

Modern Recommendation Systems rely on massive amounts of user data, such as ratings and other information on which implicit feedback can be drawn. This motivates the creation of profiles that allow for personalized recommendations. However, when this kind of information is not available, one can resort to *More Like This* recommendation (often also referred to as item–item recommendation or related item recommendation), a type of content-based recommendation that does not account for explicit user data. The goal is, given an item, to recommend other items just based on the similarity between them. This is particularly relevant when a user shows interest about a given item, but no other historic data about users or their interests is available. That is often the case when a user browses an online store, searching for a given item. In such a scenario, the properties of the items can be considered to produce improved recommendations, and some of these properties, such as the *item category*, are often directly available, while others can be retrieved or derived from the unstructured textual description of the item.

In this work, we focus on *More Like This* Mobile Application Recommendation, leveraging data provided by Aptoide,[1] a mobile application store with over 300 million users. Since, to the best of our knowledge, no other comparable collections are available, we introduced and characterized two large datasets. The first one, *Aptoide Mobile Application Dataset*, contains metadata on about five hundred thousand applications, including textual information such as names and descriptions, and global relevancy metrics. The second, *Aptoide User Information Dataset*, contains a single-day snapshot of the installed applications for over one million users.

Previous work on *More Like This* recommendation mostly focused on sparse similarity between textual fields (Colucci et al., 2016) and on how to improve this recommendation scenario with user information (Leng et al., 2018; Chen et al., 2015; Wang et al., 2017). In this

* Corresponding author at: Caixa Mágica Software, Lisboa, Portugal.
E-mail addresses: joao.coelho@caixamagica.pt (J. Coelho), diogo.mano@caixamagica.pt (D. Mano), beatriz.paula@caixamagica.pt (B. Paula), carlos.coutinho@caixamagica.pt (C. Coutinho).

[1] https://aptoide.com

https://doi.org/10.1016/j.engappai.2023.105974
Received 1 August 2022; Received in revised form 8 January 2023; Accepted 6 February 2023
Available online xxxx
0952-1976/

paper, we move from sparse to dense representations in the context of mobile applications, without using user-specific information. This work builds on our previous works on semantic search (Coelho et al., 2021b,a), which also rely on the unstructured textual information, and conclude that fine-tuned transformer-based models surpass the performance of the other existing retrieval strategies reported in the literature.

We studied the impact of semantic similarity for the task in hand, through fine-tuned transformer-based models. The proposed model recommends similar applications based on their name, description, and developer, also leveraging global relevancy metrics for ranking purposes. We compared our approach to the solution currently being used by Aptoide, which serves as our baseline. Since there are no known benchmarks for this task, a dataset containing all installed applications for each of the over one million users was used for evaluation purposes, following the intuition that, for a given application, if the applications recommended by one model M_1 co-appear more often in the installed applications of a user than the ones recommended by another model M_2, this means that M_1 is likely of higher quality than M_2.

The limitations of this co-appearance metric were addressed by resorting to user-centered tests. For this, we have created a platform where users can identify whether a set of applications is relevant or similar to a given anchor. This allowed for another evaluation mechanism, as we were able to compare the recommendations of the proposed model with the baseline.

Besides evaluating the proposed model, the user-centered tests allowed the creation of a manually annotated dataset, where for each base application, the recommendations are ranked by user-perceived similarity, which we made publicly available. To the best of our knowledge, this is the only collection of its kind in this domain. It can be used for the evaluation of future models for this task, and previous studies indicate that the usage of user-labeled similarity between items may be used during model training to enhance this type of recommendation systems.

The main contributions of this work are threefold: (i) a *More Like This* recommendation pipeline, which leverages a Transformer-based neural model to generate semantic representations for applications, (ii) a user-labeled dataset of similarity between applications, and (iii) a dataset containing a single-day snapshot of installed applications for over one million Aptoide users. The datasets were made publicly available,[2] along with a previous collection of metadata on five hundred thousand mobile applications, and the fine-tuned models.

This document is organized as follows: Section 2 covers related work on Recommendation Systems, with more emphasis on those that focus on the lack of user data. Section 3 describes the datasets used in the scope of this work. Section 4 introduces the baselines and our proposed model for recommendation. Section 5 covers our evaluation methods and results, which include tests with users. Finally, Section 6 draws conclusions and presents the future work.

2. Related work

The work here described concerns *More Like This* recommendation (also known as item–item recommendation or related item recommendation), a specific recommendation scenario where only item information is used. The remainder of this section overviews the literature on this subject, also reporting on the current state-of-the-art text ranking techniques, which we leveraged to extract the similarity between items.

2.1. More like this recommendation

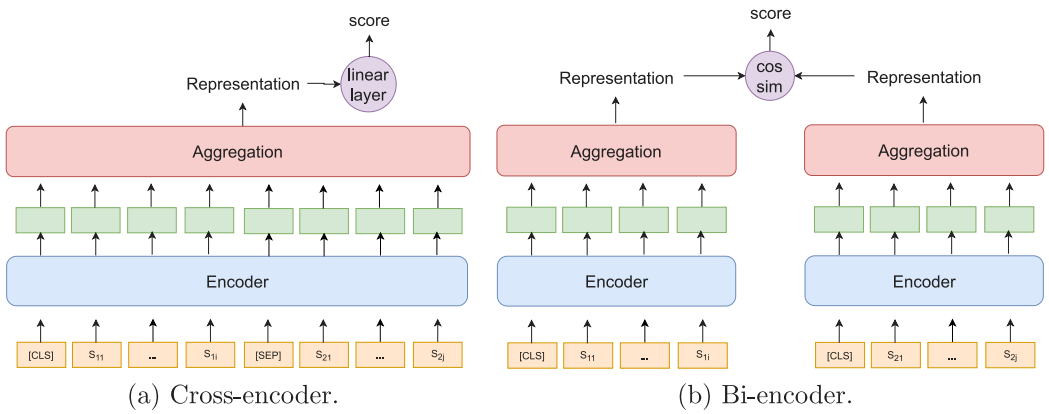
Current research on Recommendation Systems mainly focuses on collaborative-filtering approaches, made possible by the large amounts of data that is collected from users. Well-established techniques include latent factors through matrix factorization (Koren et al., 2009), an approach which has won the Netflix prize, and the system proposed and used by YouTube which introduced deep neural networks for recommendation systems (Covington et al., 2016).

Concerning the lack of user data, one of the strategies that has been studied in the literature is the usage of implicit feedback, which allows for collaborative filtering strategies, for instance by generating representations for users through auto-encoders (Shenbin et al., 2020; Liang et al., 2018; Steck, 2019; Askari et al., 2021). However, if user profiles cannot be drawn (not even implicitly), we have to rely on a recommendation based on item similarity (not to be confused with item–item collaborative filtering approaches, which try to predict ratings based on other items the user has interacted with (Lops et al., 2011)).

Previous studies have evaluated how users perceive item similarity. For instance, similarity measures, such as TF-IDF between movie textual meta-data (title, genre, cast, among others), have been considered and compared to collaborative-filtering approaches on top of the same dataset, by having users rate the perceived similarity of movie-movie pairs generated by the models (Colucci et al., 2016). Despite not relying on user data, the content-based filtering algorithm achieved comparable results to those of the collaborative-filtering models. The work of Colucci et al. (2016) was later extended by Leng et al. (2018), also reporting on the absence of benchmarks and datasets for this kind of recommendation. The main purpose of the research was to leverage the user-labeled similarity perceptions between movies obtained in Colucci et al. (2016), learning similarity functions in a supervised fashion. However, the authors also proposed content-based methods, considering more features than the previous ones (movie awards, country, among others), and different similarity measures such as BM25F (Zaragoza et al., 2004). Also, they describe a content-based model (previously evaluated, but undisclosed), which considers a combination between a movie's genre and user-contributed keywords, leveraging Solr's (open-source retrieval platform, built on top of Lucene (Bialecki et al., 2012)) *MoreLikeThis* feature. This approach surpassed the previously tested collaborative-filtering approaches. Similar studies on the same dataset were conducted, where supervised and unsupervised methods for content-based filtering were compared (Wang et al., 2017). Both works support the hypothesis that the usage of user-labeled item–item similarity for model training may increase the performance of a content-based recommendation system. Further exhaustive investigation on multiple similarity metrics between item information was conducted (Trattner and Jannach, 2020), also demonstrating the feasibility of learning similarity functions based on human annotated information. By definition, and as noted by the previous authors, unlike collaborative-filtering approaches, this type of recommendation does not suffer from cold-start problems, and as such, there are also studies accounting for the usefulness of item features to mitigate recommendation cold-start (AlRossais et al., 2021).

Closer to our work, methods for item–item semantic similarity have been proposed, where external knowledge databases were used to compare item metadata, yielding similar results to the previously described approaches (Pereira and Ferreira, 2019). Regarding mobile application recommendations, previous research focused mostly on user-centric approaches, leveraging large amounts of user data. Recent approaches rely on graph-based techniques, exploiting the graph that arises from users interacting with applications (Ouyang et al., 2021; Xie et al., 2018; Chen et al., 2020). Also, other authors proposed SimApp (Chen et al., 2015), a kernel-function based framework for computing similarity between mobile applications. It considers multiple data fields regarding the applications (title, description, images, reviews, among

[2] https://apprecommender.caixamagica.pt/resources/



(a) Cross-encoder. (b) Bi-encoder.

Fig. 1. The cross-encoder (a) and the bi-encoder (b) architectures, computing relevance scores between sentences, $S_1 = (S_{1,1}, \ldots, S_{1,i})$ and $S_2 = (S_{2,1}, \ldots, S_{2,j})$.

others), then uses multiple kernel functions to measure similarity between each data field. An online kernel learning algorithm is used to learn the optimal combination of similarity functions. Individually, the user reviews kernel is the one with the best results, which makes SimApp somewhat dependent on data that arises from user interaction with the applications. Bhandari et al. (2013) have worked on the problem of recommending applications that users have never interacted with, leveraging an application–application similarity graph built using TF-IDF similarity between textual fields.

Finally, there are studies that account for the suitability of neural features in other recommendation scenarios. For instance, convolutional neural networks have been used to extract features for image-based recommendation (Messina et al., 2019), and pre-trained neural language models have been used to generate embeddings for news recommendations (Wu et al., 2021).

2.2. Text ranking

We model our problem as a full-retrieval task, i.e., given one application (henceforth referred to as query), get the top ranked ones from a potentially very large collection (henceforth referred to as documents), based only on their textual information. Currently, neural methods achieve state-of-the-art results on tasks such as this one, that include ranking short texts according to relevance towards a query (Lin et al., 2020). Most approaches are based on the usage of Transformer-based (Vaswani et al., 2017) neural language models, either following bi-encoder or cross-encoder architectures, represented in Fig. 1. Cross-encoders leverage language models such as BERT (Devlin et al., 2019) to directly generate representations for concatenations of a query and a document. This architecture is mostly used for re-ranking tasks (i.e., re-rank the top-N items retrieved by more efficient methods). For instance, a feed-forward layer, trained together with the transformer model, is used to predict a relevance score, given a representation for the concatenation of the query and the document (Nogueira and Cho, 2019). Conversely, bi-encoders represent queries and documents independently, which allows the offline indexing of individual document representations through methods that support the fast execution of maximum inner product searches (Johnson et al., 2017). As such, the bi-encoder architecture is more efficient than cross-encoders, hence being used for full-ranking tasks, i.e., retrieve the top-N items from the whole collection. These models are often trained by using loss functions that promote high similarity between positive query-document pairs, and a low similarity between negative ones (Reimers and Gurevych, 2019; Qu et al., 2021). However, query-agnostic training methods have also been proposed (Gao and Callan, 2021; Ram et al., 2021).

2.3. Proposed approach – key differences

Most of the recommendation systems focus on collaborative-filtering approaches that use large amounts of historical data, and sometimes rely on implicit feedback when no other data is available. Concerning the *More Like This* recommendation, the literature is scarce, and even more scarce when no user interaction data exists and only item information is used. We have also mentioned different text ranking techniques, which can be used to extract the similarity between items based on their textual descriptions, and we have concluded that the most recent approaches are based on transformer models.

The key differences between this work and the aforementioned ones are as follows: (i) we tackle the specific case of mobile app recommendation where **no user-specific information is available for recommendation**; (ii) we use a Transformer-based model to generate **contextual application embeddings**, which was trained in a query-agnostic fashion; and (iii) we introduce **two novel benchmarks** regarding mobile applications for training and evaluation, as well as user-centered tests.

3. Datasets

This section describes the datasets used in the scope of this work, namely the *Aptoide Mobile Application Dataset*, which contains metadata on mobile applications, including the textual information and relevancy metrics, and the *Aptoide User Information Dataset*, which contains information regarding applications installed on the mobile devices of over 1 million users.

3.1. Aptoide mobile application dataset

In previous work (Coelho et al., 2021b,a), we have compiled a large dataset with metadata on around 500 thousand applications through Aptoide's API. It includes information such as global relevancy metrics (downloads, ratings, average rating), textual fields (title, description, category), icons, and other data that is not relevant for the scope of this work. Table 1 shows some of the information for a sample application (Instagram). Note that the relevancy metrics are static and relative to the time of the data retrieval (April 14th, 2021). This dataset was used for fine-tuning neural language models, given the large amounts of textual data. Only the name and descriptions of the applications were considered for this purpose. Keywords were discarded since there was no information on their generation process. As for the news, their usage was not consistent for the majority of the applications, and as such, were not considered. Fig. 2 is an example of an application's description containing 111 words distributed in multiple sentences.

The relevancy metrics were used for ranking purposes in the recommendation pipeline. Fig. 3 displays the number of applications within a

Table 1
Information within Aptoide Application Dataset for application "Instagram".

title	Instagram
description	Bringing you closer to the people and things you love...
downloads	92071704
average rating	4.4
total ratings	14076
last update	2021-04-02 09:16:33
icon	*url*
keywords	Instagram; Android; Social
screenshots	*url*
news	We have new features and improvements! Update...
category	Social

Don't Starve: Pocket Edition, brings the hit PC game enjoyed by over 6 million players to Android.
Now you can experience the uncompromising wilderness survival game full of science and magic on the go! Play as Wilson, an intrepid Gentleman Scientist who has been trapped and transported to a mysterious wilderness world. Wilson must learn to exploit his environment and its inhabitants if he ever hopes to escape and find his way back home.
Enter a strange and unexplored world full of strange creatures, dangers, and surprises. Gather resources to craft items and structures that match your survival style. Play your way as you unravel the mysteries of this strange land.

Fig. 2. Textual data regarding the application entitled "Don't Starve: Pocket Edition".

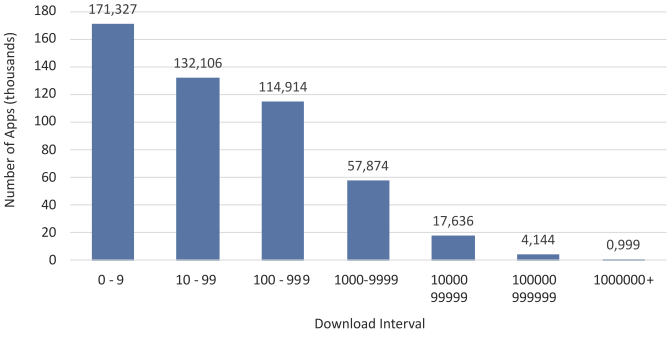


Fig. 3. Number of applications with an interval of download values.

total download interval, showing that the majority of the applications have very few downloads. Also, 414,053 of the applications within this dataset have never been rated (i.e., their *total ratings* feature is 0). For the remaining ones, the rounded average rating distribution is depicted in Fig. 4. Finally, Fig. 5, shows the age (in years) distribution of the applications. Overall, this suggests that this is a recommendation domain where the cold-start problem could be visible for recommendation systems that rely solely on user feedback, since the majority of the applications are very recent and have few to no ratings, further motivating our approach for *More Like This* recommendation, leveraging the textual data within the applications to infer semantic similarity.

For future reference, although this information will not be used in this work, the dataset also includes 147 unique user-queries (e.g., queries that could be posed to a mobile application search engine), each labeled with 5.2 relevant applications on average.

An exploratory analysis of the applications have shown that the vast majority of the applications were not being updated frequently, were rarely downloaded, or were associated with low ratings or with a very few number of ratings. Since such applications are not of much relevance for a recommendation system, we have decided to create a smaller subset of applications,[3] restricted based on the number of days since the last update, the number of downloads, based on the ratings, and also including a final manual validation. Hence, the experiments

[3] https://apprecommender.caixamagica.pt/resources/

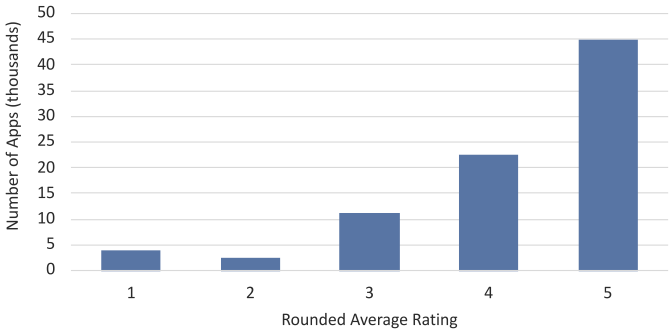


Fig. 4. Rounded average rating distribution.

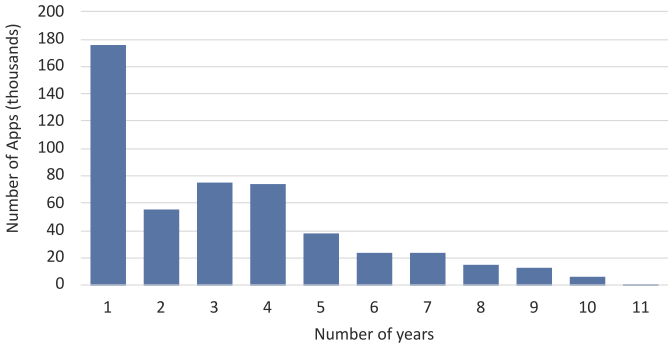


Fig. 5. Application age (in years) distribution.

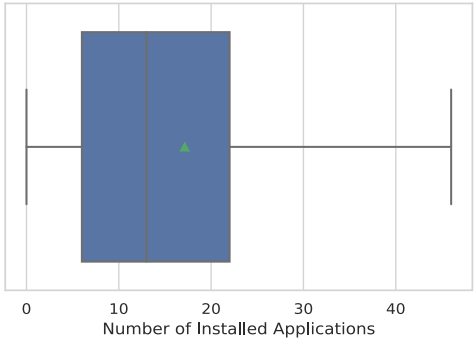


Fig. 6. Distribution of the number of installed applications per user. Green triangle represents the mean value.

described in the scope of this work use this smaller subset of the dataset, containing approximately 6000 applications, from which the recommendations are chosen.

3.2. Aptoide user information dataset

This second dataset contains information concerning the installed applications for 1,034,104 users. It comprises a single-day snapshot of active users and their installed applications. Each user is identified by a hashed identifier, complying with Aptoide's data protection policy. Applications are also represented by an identifier, which is consistent with the one used in the *Aptoide Mobile Application Dataset*.

Fig. 6 depicts the distribution of the number of installed applications per user, where the outliers were removed for visualization purposes. In average, each user has 17 installed applications. An analysis of the applications themselves was also conducted. Table 2 shows the top 10 installed applications in this snapshot, along with other available relevance metrics.

Table 2
Top 10 installed applications within the snapshot, along with other information available in the *Aptoide Mobile Application Dataset*.

Application name	Installations	Average rating	Days since last update	Publish date
WhatsApp Messenger	550209	4.19	12	2012–10–24
Facebook	490676	4.16	12	2013–02–15
Instagram	356237	4.40	12	2014–08–31
Messenger	314794	4.32	12	2014–07–25
Google Duo	198639	4.49	13	2016–09–22
FactoryCamera	187795	3.5	403	2019–03–25
MS PowerPoint	140059	4.35	12	2021–02–15
Foundation	134651	3.39	1124	2018–03–17
Dictionnaire	131265	3.36	111	2018–06–11
Netflix	129232	3.73	13	2018-02-22

User: a45500cf4b6b2200c1682154cd2a2eb09ae1dc9d6b9c31194233a50b6ac85e0e
Installed Apps: ['com.miui.huanji', 'com.instagram.android', 'com.miui.global.packageinstaller', 'com.miui.aod', 'com.miui.cit', 'com.miui.android.fashiongallery', 'com.google.android.apps.wellbeing', 'com.google.android.dialer', 'com.netflix.mediaclient', 'com.goodix.fingerprint.setting', 'com.mi.android.globalminusscreen', 'com.facebook.lite', 'cm.aptoide.pt', 'com.miui.newmidrive', 'com.whatsapp', 'com.miui.gallery', 'com.netflix.partner.activation', 'com.android.vending', 'com.tencent.ig', 'com.google.android.play.games', 'com.miui.player', 'com.google.android.apps.maps', 'com.miui.cloudbackup', 'com.google.android.gms', 'com.miui.hybrid.accessory', 'com.miui.calculator', 'com.miui.cleanmaster', 'com.xiaomi.scanner']

Fig. 7. Sample of the *Aptoide User Information Dataset*, mapping an anonymized user to its installed applications.

Since this work focuses on developing methods for recommendation without leveraging individual user data, this dataset for evaluation purposes only in the scope of the work here reported. Fig. 7 shows a sample from this dataset.

4. Recommendation of mobile applications

This section introduces the *More Like This* recommendation systems currently in use by Aptoide, here considered as the baseline, and proposes a multi-criteria alternative based on semantic similarity and global relevancy judgements.

4.1. Current Aptoide models

Currently, Aptoide has two solutions for *More Like This* Recommendation. The first one (Aptoide M1), for a given application, recommends the seven applications of the same category with higher number of downloads. The second one (Aptoide M2) is undisclosed, but it can be used through API endpoints in order to produce recommendation results for a given application.

While there are no previous studies accounting for the quality of the aforementioned systems, we will use them as baselines since (i) there are no other known studies for this specific task, and (ii) Aptoide is a large store with millions of users, which makes it safe to assume that the models tend to be optimized towards conversion.

4.2. Multi-criteria semantic model

In previous work, we proposed RoBERTapp (Coelho et al., 2021b,a), a fine-tuned version of $\text{RoBERTa}_{\text{base}}$ (Liu et al., 2019) over large amounts of mobile applications textual data. We briefly cover RoBERTapp fine-tuning, and then discuss its usage within a multi-criteria semantic model for recommendation.

4.2.1. RoBERTapp

Previous studies showed that out-of-the-box BERT-like embeddings are unsuitable for semantic-similarity tasks (Reimers and Gurevych, 2019). As such, the $\text{RoBERTa}_{\text{base}}$ model was fine-tuned in two tasks. First, masked language modeling was used over the name and description of approximately four hundred thousand applications with English text. Then, the model was trained on a semantic similarity task. We leveraged the same dataset that was used for the masked language modeling objective, namely application names, descriptions, and categories. For this sort of training, a large set of queries labeled with relevant applications would be useful, but this is not available in the dataset. As such, the semantic training task consisted in distinguishing between real and fake descriptions for a given synthesized query, which is obtained by concatenating an applications name with its category. For instance, consider the application *instagram*. During training, the query *instagram social* would be compared to the real description (Table 1) and to randomly sampled fake descriptions (for instance, the one showed in Fig. 2 could be used). The standard binary cross entropy loss is used to enforce high similarity between positive pairs, and low similarity between negative ones:

$$L_{BCE} = -\sum_{d \in D_q^+} \log(\text{score}(q,d)) - \sum_{d \in D_q^-} \log(1 - \text{score}(q,d)) \,. \tag{1}$$

In the previous equation, D_q^+, D_q^- are the sets of positive and negative descriptions for query q, respectively, provided within the same training batch. To obtain the $\text{score}(q,d)$, i.e, an estimate of the relevancy of description d to query q, the representations for q and d are generated by mean pooling of the token vectors of the last hidden layer, and the cosine similarity is used to compute the score. Each training batch contained 10 pairs of queries associated with the real description. Batch-wise negative pairing was applied, by using the relevant description of a given query as negatives for the others, increasing the effective training data. The model was fine-tuned under the default RoBERTa configuration, using the *Aptoide Mobile Application Dataset* in a 90–10 split for validation, during one epoch, a learning rate of 2e−5 with a linear warmup scheduler during the first 5000 iterations, using AdamW (Loshchilov and Hutter, 2019) as the optimizer.

The final model outperformed previous approaches (lexical retrievers and word-embeddings) for semantic search tasks, for instance on the dataset used by Park et al. (2015), where the task consisted in retrieving and ranking applications given non-specific queries (i.e., *social networks*).

This work uses the model in a bi-encoder style to compute similarity between applications, by taking the cosine similarity of representations for the names, descriptions, and developers. Representations are generated by mean pooling of the token vectors of the last hidden layer. This architecture is particularly well-suited for this scenario, since all textual information regarding applications can be encoded offline and indexed, resulting in fast run-time searches.

4.2.2. Recommendation pipeline

As explained in Section 3.1, the experiments here described use a subset of approximately 6000 applications extracted from Aptoide's Application Dataset, from which the recommendations are chosen.

Assuming that an user selects the application a_1, the recommendation process for that application is as follows: First, the similarity between a_1 and the other applications is computed through cosine similarities of model generated embeddings ($E[.]$):

$$\begin{aligned} \text{sim}(a_1, a_i) = {} & w_n \cos_{\text{sim}}(E[a_{1,n}], E[a_{i,n}]) + \\ & w_d \cos_{\text{sim}}(E[a_{1,d}], E[a_{i,d}]) + \\ & w_{dev} \cos_{\text{sim}}(E[a_{1,dev}], E[a_{i,dev}]) \,, \end{aligned} \tag{2}$$

where $a_{j,n}$, $a_{j,d}$, $a_{j,dev}$ correspond to the *name*, *description* and *developer* textual representation of application j, respectively. Combination weights, namely w_n, w_d and w_{dev} are also considered for each feature.

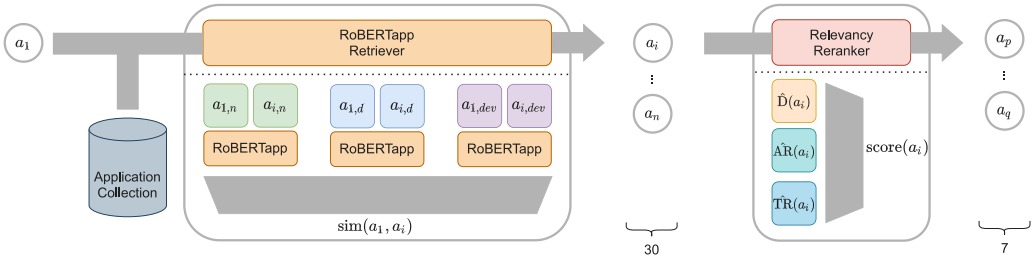


Fig. 8. Illustration of the recommendation pipeline for application a_1. First, the neural retriever selects the top-30 similar applications, which are then re-ranked based on relevancy metrics, and the top-7 is chosen.

Then, the top-30 scoring ones are chosen, and re-ranked by relevancy score, which is computed by considering the number of downloads, average rating and total ratings:

$$\text{score}(a_i) = w_{down}\,\hat{\text{D}}(a_i) + w_{ar}\,\hat{\text{AR}}(a_i) + w_{tr}\,\hat{\text{TR}}(a_i)\,, \tag{3}$$

where w_{down}, w_{ar}, and w_{tr} are combination weights, and $\hat{\text{D}}(.)$, $\hat{\text{AR}}(.)$, and $\hat{\text{TR}}(.)$ are functions that return a normalized value for downloads, average rating, and total ratings, computed as follows:

$$\hat{\text{D}}(a_i) = \frac{\log \text{D}(a_i) - \log \min_{a\in A} \text{D}(a)}{\log \max_{a\in A} \text{D}(a) - \log \min_{a\in A} \text{D}(a)}\,, \tag{4}$$

$$\hat{\text{AR}}(a_i) = \frac{\text{AR}(a_i)}{5}\,, \tag{5}$$

$$\hat{\text{TR}}(a_i) = \frac{\text{TR}(a_i) - \min_{a\in A} \text{TR}(a)}{\max_{a\in A} \text{TR}(a) - \min_{a\in A} \text{TR}(a)}\,, \tag{6}$$

where A is the subset of applications from which the recommendations will be chosen. The top-7 applications in the ranked list are the final recommendations. Fig. 8 depicts the whole pipeline.

Several additional adjustments can be considered for production scenarios, such as accounting for trendiness (e.g. use the number of downloads in the last 7 days instead of the current all-time downloads value).

5. Evaluation strategies and results

This section presents the three evaluation strategies and the corresponding results. First, an automatic evaluation based on installation co-occurrences is introduced. Then, in order to better understand the achieved results, we qualitatively analyze the model recommendations and representations. Finally, we conduct user-centered tests to enforce the results, also yielding useful data for future evaluations.

5.1. Automatic evaluation

Given the lack of benchmarks for this specific task, we leverage the *Aptoide User Information Dataset* in order to evaluate and compare the proposed model with the method currently being used by Aptoide. This is achieved by computing the co-occurrence of installed applications. While we did not find evidence on co-occurrence being used for evaluation purposes, some work has been done on using item co-occurrence information for recommendation systems (Yagci et al., 2017; Zhang et al., 2021; Qin et al., 2021). Hence, we made the dataset available to facilitate future research on this topic.

As for the metric, let R_{M_i,a_j} be the set of recommend apps by model M_i for application a_j. Let I_{u_k} be the set of installed applications for user u_k, and U^{a_j} the set of all users who have a_j installed. The recommendation co-occurrence can be computed as follows:

$$\text{score}(M_i, a_j) = \frac{\sum_{u_k \in U^{a_j}} \frac{|R_{M_i,a_j} \cap I_{u_k}|}{\min(|R_{M_i,a_j}|, |I_{u_k}|)}}{|U^{a_j}|}\,. \tag{7}$$

Given a set of applications, A, we compute the average score for all applications as follows:

$$\text{score}_{\text{avg}}(M_i, A) = \frac{\sum_{a_j \in A} \text{score}(M_i, a_j)}{|A|}\,. \tag{8}$$

It is worth noting that this value should be properly interpreted. A score of 0 between two applications means that no user has them installed together. Conversely, a score of 1, means that all the users that have installed one of them, have also installed the other, and vice-versa. None of these scores convey a strong recommendation hypothesis, since in the first example no user is likely to install the recommended application, and in the second example, users are likely to have the recommended application installed already. Nonetheless, we consider that for intermediate score values, if $\text{score}_{\text{avg}}(M_1) > \text{score}_{\text{avg}}(M_2)$, then M_1 is likely a superior model. Even though this metric does not directly consider the similarity between applications, we still believe this metric is relevant, since it reflects the actual usage of applications by the users, which is of upmost importance for this kind of recommendation systems. As such, we start by validating the model on this data, by tuning the multiple combination weights introduced in Section 4.2.2, thus allowing us to draw conclusions related with each one of the textual fields and relevancy metric.

As for textual fields, the validation results show that the best combination is $w_n = 0.1$, $w_d = 0.5$, and $w_{dev} = 0.4$. The description plays an important role, probably due to the fact that the RoBERTapp model was fine-tuned on a task which tried to distinguish between real and fake descriptions. The textual information that can be extracted from the developer is rather limited (it is more important for clustering), but giving more weight to this field increases the results. This may indicate that users tend to install multiple applications from the same developer. Increasing the weight given to the name did not produce better results. We argue that this may be due to the fact that names are usually small, and all of its information is contained and contextualized within the description.

As for relevancy metrics, the combination $w_{down} = 0.8$, $w_{ar} = 0.1$, and $w_{tr} = 0.1$. was the one that achieved the highest results. The number of downloads was the most influential parameter, as we expected, given the task's nature (i.e., the number of downloads will play a more significant role in application co-occurrences than the others). While the weights related to textual fields are model-dependent (and should probably only need to be changed drastically if the training task of the model changes), these relevancy weights can be tuned differently for different purposes. For instance, in a deployed solution (e.g., integrated in Aptoide's services), these weights could be changed in order to promote more recent applications rather than well-established ones.

Finally, we computed the co-occurrence scores for each one of the models, given by Eq. (8), on a test containing 35 applications, randomly sampled from the most downloaded ones. Table 3 shows the corresponding results, considering individual results for sub-groups of applications of the same category (as given by Aptoide), and the group of all the applications. Overall, the proposed model achieved a score of 14.4%, clearly outperforming both Aptoide current solutions in terms of installation co-occurrences. These promising results led us

Table 3
Scores achieved by both Aptoide approaches and the proposed model. Results shown for all applications in the test set, and for subsets of applications in different categories.

	Aptoide M1	Aptoide M2	Proposed model
Communication	17.7%	13.9%	**19.9%**
Social	17.9%	7.7%	**36.4%**
Productivity	1.7%	0%	**23.0%**
Games	2.8%	0.8%	**3.3%**
Tools	~0%	~0%	**3.2%**
Others	2.7%	1.3%	**14.5%**
All Applications	5.5%	3.1%	**14.4%**

to compare the models even further, by manually analyzing a number of actual recommendations provided by each one of the models. The remainder of this section describes that process and the corresponding achievements.

5.2. Qualitative evaluation

The previous results show that the proposed model outperforms both Aptoide current solutions, in terms of installation co-occurrences. Now, we try to identify the differences by manually comparing the actual recommendations of the proposed model with the Aptoide best scoring approach.

Fig. 9 shows the recommended applications by the two models for four sample base applications of different categories: *Garena Free Fire*, *Duolingo*, *Avast*, and *Dropbox*. Recall that Aptoide M1 works by retrieving the top-7 downloaded applications from the same category of the base. The results shown for this model were obtained on December 19, 2021, but it is important to refer that the recommendations may differ in the future, since the model is not static. It seems that Aptoide's categorization is somewhat broad, since some applications are grouped together even though they serve completely different purposes (e.g., Dropbox, a cloud backup application, and Mobizen, a screen recorder). Other problem inherent to this recommendation strategy is that all applications of the same category will show the same seven widely-used recommendations, which we argue to hinder conversion rates. As for the proposed model, we can see that the vectorial representations generated by RoBERTapp seem to capture the semantic context of applications (e.g., for Dropbox, the recommended applications are cloud related).

To further understand the results, we looked deeply into these representations. Here, we considered an application to be represented by the average of the embeddings for its title and description. Fig. 10 shows a t-SNE plot (van der Maaten and Hinton, 2008) of ten applications from three different categories. The semantic capabilities of the model seem to be able to capture intra-category similarities, as there is a clear separation between them. Also, even for the streaming services, a separation between music and movies can be observed. This suggests that the proposed model generates representations that provide meaningful clustering of applications.

5.3. User-centered tests

Our automatic evaluation procedure differs from the commonly used ones, for instance, when computing the RMSE over a test set of user-labeled app ratings. While our proposed evaluation method provides valuable insights, one can argue that co-occurrence may not be directly related to relevancy, nor can it be assumed that two applications are similar because they co-appear on a given user's phone. Hence, we resorted to tests with users. Besides being able to further evaluate the proposed model, by directly comparing it to Aptoide's current solution, previous studies suggest that the collected user-similarity labeled items may be useful for future evaluations.

The platform architecture comprises the recommendation mechanisms, which were built as services, and a front-end, which was implemented leveraging said services. To further motivate studies in this area, we made the code that supports this platform open-source.

For the tests, we had the support of Aptoide, in the sense that real users were notified on their smartphones to answer the tests. A total of 1262 English-speaking users completed the survey. When starting the tests, the users were presented a welcome message including the rules. The users were prompted to indicate their name and country for a simple demographic study.

5.3.1. Testing platform

The mobile interface is shown on Fig. 11. The test was divided in 10 steps, and in each one, one base application (randomly sampled from the test set) was given, along with seven recommendations made by each model (ours and Aptoide's). The user could then select which recommended applications were relevant for a user that installed a base application, in his/her opinion, also being able to click on each application to get more information about it, if necessary (Fig. 12, a). If both models return a given application, only one instance will be presented in the visualization. In the end, for engagement purposes, the testing platform presents a number of statistics regarding which model was preferred by the user.

To minimize any possible bias, the user does not know which model recommended which applications, and the information regarding each application only shows its description and icon, i.e., ratings and other relevancy metrics were omitted. Also, as there was no direct contact between us and the users, rules were made available throughout the whole test (Fig. 12, b).

5.3.2. Statistical relevance and results

As previously mentioned, a total of 1262 users fully completed the quiz. Overall, the users found a total of 23,352 applications recommended by the proposed model as relevant, and 14,984 relevant ones recommended by Aptoide's current approach. A Fisher's exact test ensured the statistical significance of the improvement, yielding a p-value of $p < 0.001$. Moreover, we conclude that approximately 27% of the recommendations given by our proposed model are relevant, a significant improvement over the current Aptoide solution, which yields only about 17% of relevant recommendations.

We now present some recommendation comparisons along with the number of relevant votes for each application, i.e., how many users deemed said applications relevant for the base one. Fig. 13 shows two examples where the proposed model performed well, namely (a) the one with the highest difference in votes when compared to Aptoide's, and (b), the one with the highest overall number of votes. Conversely, Fig. 14 shows two examples where the proposed model did not perform as well, namely (a) the one with the smallest difference in votes when compared to Aptoide's, and (b), the one with the smallest overall number of votes. This motivates the fact that while the proposed model is able to capture similarities between applications, there is still room for improvement. For example, for *Skin Editor for Minecraft* (Fig. 13, b), while the proposed model does capture the context of the application, it only performs well in comparison to Aptoide's because the latter seems to be due to some miscategorization. For *Facebook*, (Fig. 13, a), the recommendations are rather similar, however the users deemed ours more relevant. It is worth noting that Aptoide's will be the same for all social networks, while ours is more prone to change. Regarding *PUBG Mobile Lite*, (Fig. 14, b), the proposed model provides clearly worse recommendations. We hypothesized that the proposed model was not good at encoding games textual information, given this and the score achieved in Table 3. However, this does not generalize for all games, since, for instance, in Fig. 9, for *Garena Free Fire* (a game of the same genre), the proposed model provides more meaningful recommendations when compared with *PUBG Mobile Lite* (also, note that Aptoide's recommendations do not change from *PUBG Mobile Lite* to *Garena Free Fire*). Nonetheless, this may indicate that it might be a

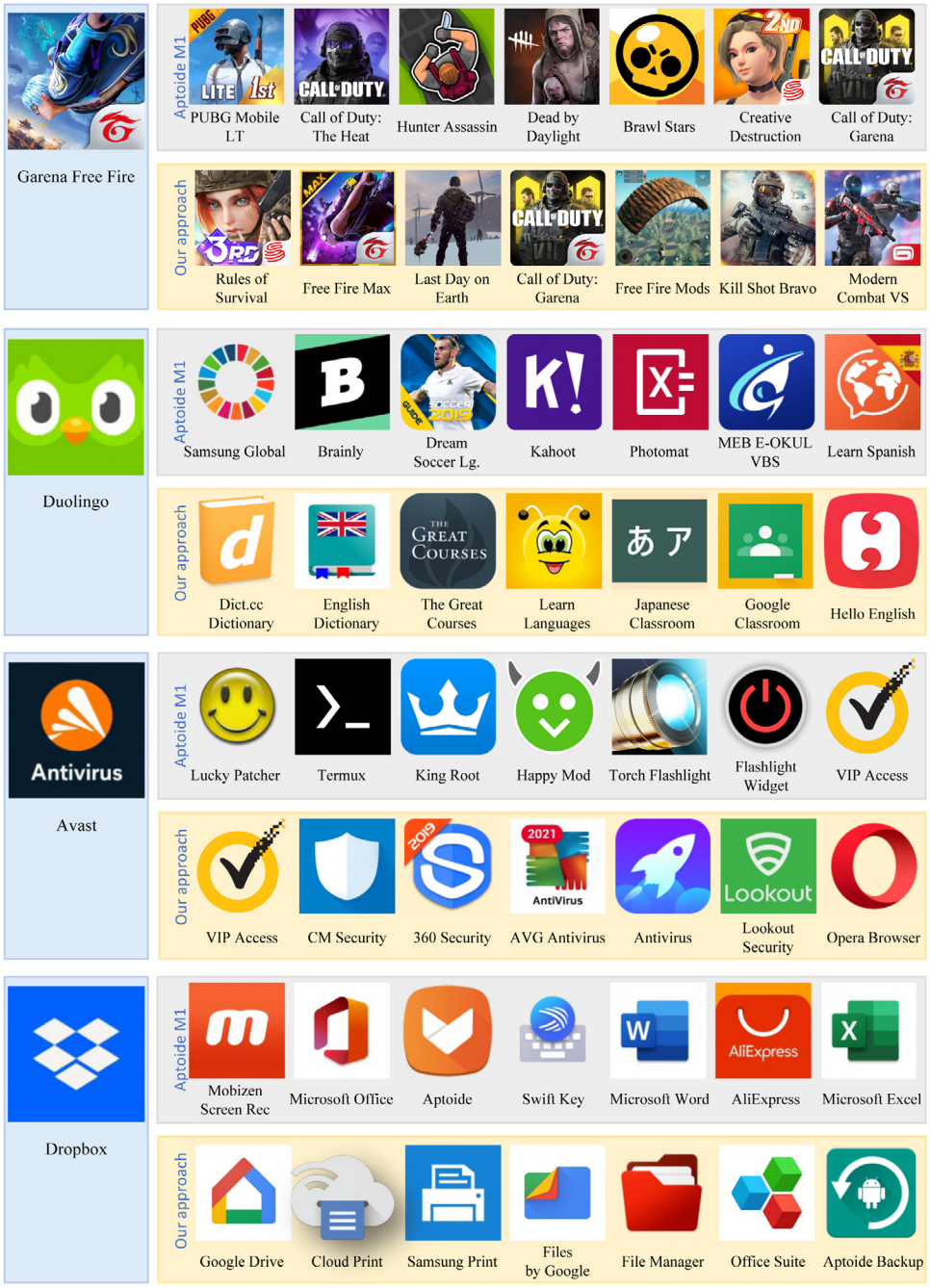


Fig. 9. Recommendations for "Garena Free Fire", "Duolingo", "Avast", and "Dropbox".

good idea to have a model fine-tuned for games only, and another for the remaining applications.

Altogether, the results of the user-centered tests confirmed the usefulness of including semantic information when recommending similar applications. Also, we were able to extract a dataset of 35 applications, each associated with up to 14 applications (due to both models recommending the same application, some may not have 14 unique recommendations) ranked in terms of similarity/relevancy towards the base one. As such, we now present evaluation results on top of this novel dataset, using multiple methods to rank the 14 applications for each of the 35 base ones, considering the user votes as ground truth. RoBERTapp was compared with three baselines, namely the number of downloads, a lexical baseline (BM25 (Robertson and Zaragoza, 2009)), and $\text{RoBERTa}_{\text{base}}$.

For each of the base applications, its textual fields (name and description) were compared against the ones of the 14 candidate applications. For BM25, the name and descriptions were concatenated, with the score being computed as follows:

$$\text{score}_{\text{BM25}}(q,d) = \sum_{i \in q} \text{idf}(i) \times \frac{\text{tf}(i,d) \times (k_1+1)}{\text{tf}(i,d) + k_1 \times \left(1 - b + b \times \frac{|d|}{\text{avgdl}}\right)}. \quad (9)$$

In the previous equation, b and k_1 are hyperparameters (in this case, tuned to the values of 0.75 and 1.5, respectively), avgld is the average length of the passages in the collection, $\text{tf}(i,d)$ is the frequency of term i within passage d, and $\text{idf}(i)$ is the inverse document frequency for term i.

As for $\text{RoBERTa}_{\text{base}}$ and RoBERTapp, the models were used to compute representations for the name and description separately, and

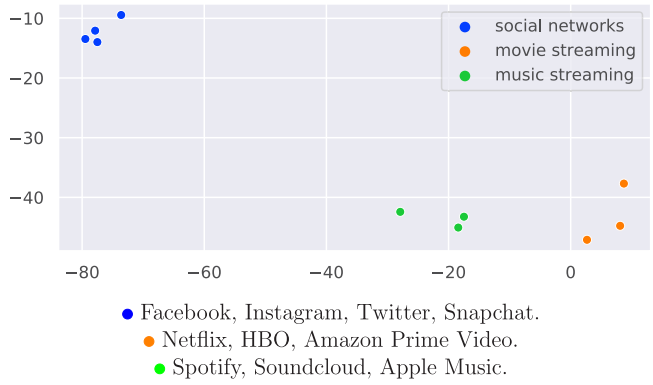
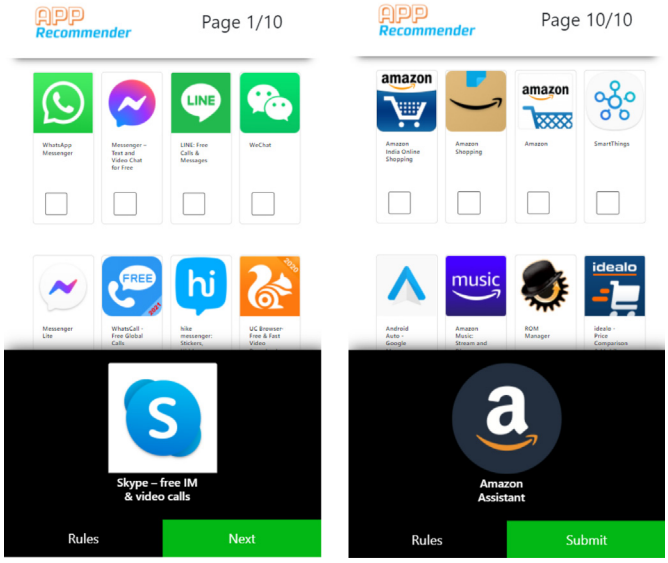


Fig. 10. T-SNE plot of the representations of 10 applications. Color represents their original categories.



(a) First base application (Skype) (b) Last base application (Amazon Assistant)

Fig. 11. Mobile interface of the user-centered tests, shown for the first (a), and for the last (b) application of an example test. On the bottom, a menu showing the application for which the recommendations are being presented, along with "Next" and "Rules" buttons. The former progresses through the test which comprises 10 applications, while the latter shows the rules of the test. The page of the final application contains a "Submit" button instead of the "Next".

a score was computed through weighted cosine similarity of model generated embeddings ($E[.]$):

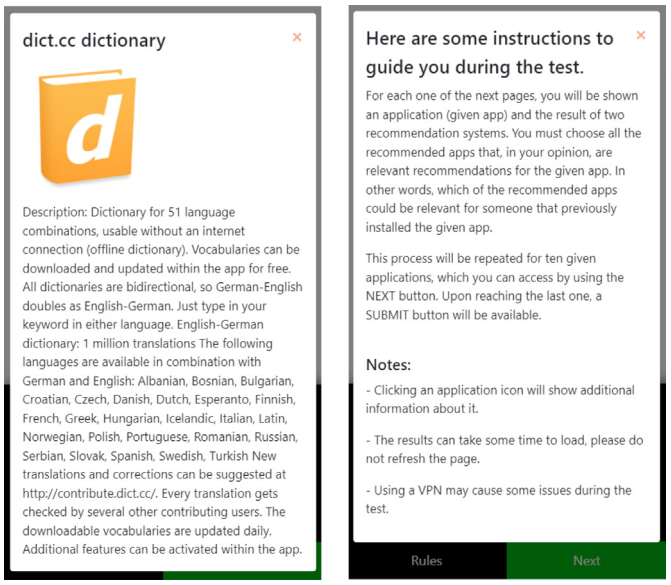
$$\begin{aligned}\text{sim}(a_{\text{base}}, a_{\text{candidate}}) = w_n \cos_{\text{sim}}(E[a_{\text{base},n}], E[a_{\text{candidate},n}]) + \\ w_d \cos_{\text{sim}}(E[a_{\text{base},d}], E[a_{\text{candidate},d}]) ,\end{aligned} \tag{10}$$

where $a_{j,n}$, $a_{j,d}$, correspond to the *name* and *description* textual representation of application j, respectively. Combination weights, namely w_n, w_d, were tuned to 0.3 and 0.7, respectively, since the results from Section 5.1 showed that giving more weight to the description is beneficial, possibly due to RoBERTapp's training task.

Table 4 shows the results using the Normalized Discounted Cumulative Gain (NDCG) as the evaluation metric, which takes the full order of the item list and graded relevance into account:

$$\text{DCG@}k = \text{R}(1) + \sum_{i=2}^{k} \frac{\text{R}(i)}{\log_2(i)} , \tag{11}$$

$$\text{NDCG@}k = \frac{\text{DCG@}k}{\text{IDCG@}k} , \tag{12}$$



(a) Application information (b) Rules

Fig. 12. Information for a specific application that can be accessed by clicking on it (a), and rules that are available throughout the whole test (b).

Table 4
NDCG@{6,10,14} achieved by ranking the 14 recommendations for each of the 35 base applications with multiple approaches, considering the user votes as ground truth.

	NDCG@6	NDCG@10	NDCG@14
Downloads	0.6236	0.7378	0.7996
BM25	0.7769	0.8411	0.8857
$\text{RoBERTa}_{\text{base}}$	0.7421	0.8191	0.8713
RoBERTapp	**0.8728**	**0.9140**	**0.9303**

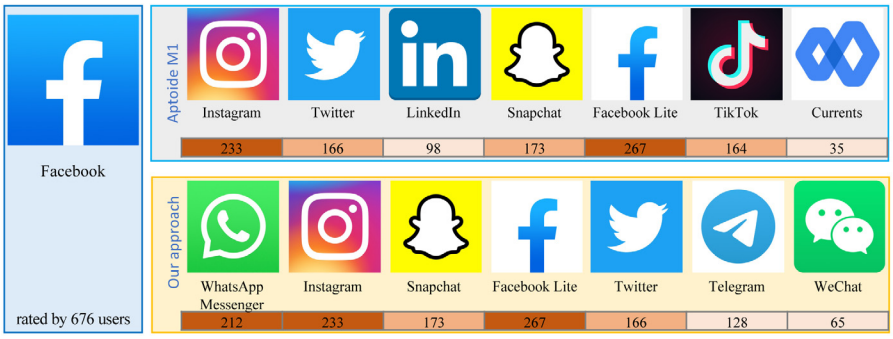
where R(i) is a function that returns the relevance value of the passage at rank i. The index of the passage up to which the ranking is considered is represented by k. The DCG is normalized with the ideal DCG (IDCG), i.e., the DCG of a perfectly sorted result.

Overall, the RoBERTapp model was able to outperform all the other tested approaches, enforcing the usefulness of semantic similarity in this context. Also, the $\text{RoBERTa}_{\text{base}}$ model produced worse results when compared to the lexical baseline, showing the adequacy of our fine-tuning strategy for the specific domain of mobile applications.

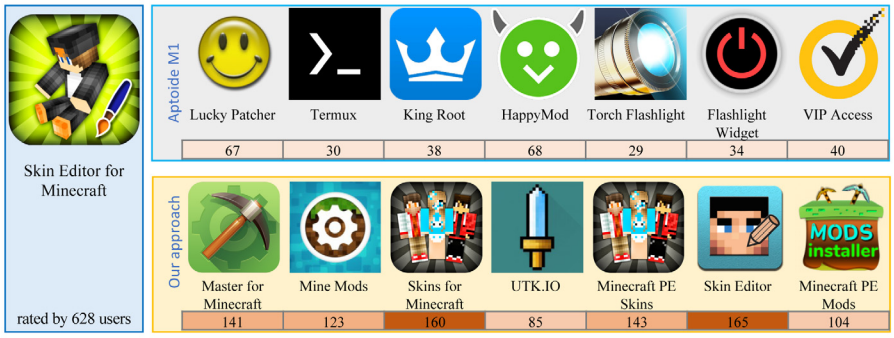
6. Conclusions and future work

Recommendations based on direct user input are not always possible, and may even constitute a bigger problem in the future, since user privacy is a trending issue. As such, this work studies semantic similarity for the recommendation of mobile applications, focusing on a *More Like This* recommendation, i.e., given an application, recommends similar ones based solely on their textual properties, namely the name, description, and developer, without relying on user information. More specifically, we leveraged semantic similarity through neural language models and global relevance statistics when recommending applications.

This paper proposes a pipeline for *More Like This* recommendation, built on top of a Transformer-based encoder, which was fine-tuned on large amounts of textual data regarding mobile applications. The pipeline retrieves the candidate similar applications, considering the textual representations of their name, description, and developer. Then, the retrieved applications are re-ranked by global relevancy metrics, namely downloads, total ratings, and average ratings. We have proposed two evaluation strategies in order to properly compare our

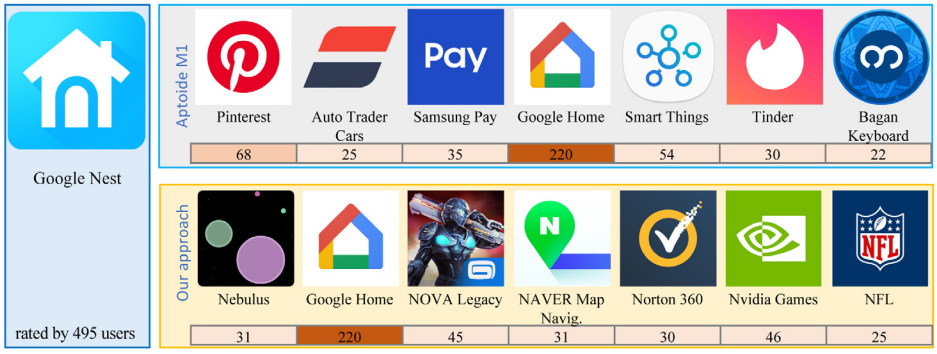


(a) Recommendation comparison for Facebook

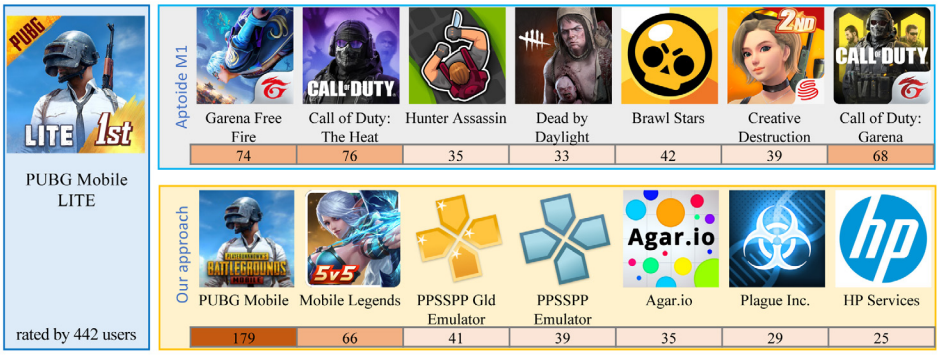


(b) Recommendation comparison for Minecraft Skin Editor.

Fig. 13. Two applications where the proposed model performed well in the user-centered tests, (a) the one with highest difference in relevant votes against the Aptoide model, and (b), the one with the higher number of relevant votes.



(a) Recommendation comparison for Google Nest



(b) Recommendation comparison for PUBG Mobile

Fig. 14. Two applications where the proposed model performed poorly in the user-centered tests, (a) the one with smallest difference in relevant votes against the Aptoide model, and (b), the one with the lowest number of relevant votes.

proposed model with two other approaches currently in use by Aptoide. The first evaluation strategy, fully automatic, consists of computing co-occurrences of applications over the *Aptoide User Information Dataset*. The second approach is based on user-centered tests, leveraging Aptoide's massive user base, where more than 1 thousand Aptoide users evaluated a series of recommendations from two different models. Both evaluation strategies showed that our proposed semantic approach outperforms the existing Aptoide approaches, motivating further research, proper benchmarking, and the inclusion of semantic cues on item–item recommendation systems.

Concerning scalability, we strongly believe that our proposal does not pose major scalability issues. While the fine-tuning stage takes a significant computational effort, it can be performed periodically and offline, without a significant loss in performance. The model can then be used to produce the top similarity scores for each one of the applications, which can be stored as a static list associated with the given application.

While the related work on this specific subject is rather scarce, previous studies indicate the lack of data (either for learning or for evaluation) as one of the main concerns. As such, we disclosed a dataset of over 1,000,000 users and their installed applications (*Aptoide User Information Dataset*), and a dataset gathered through user-centered, containing a range of applications labeled with similar/relevant applications. For the latter, we provide evaluation results for the task of ordering the recommendations for each base application, considering the order given by total user votes as ground-truth.

As for future work, we observed that results given by our RoBERTapp model, trained on textual data regarding all types of applications, showed a degree of variation for different application categories. This way, training a model specifically for each category may improve the results even further. Also, we are considering using the data on Aptoide's User Information Dataset to draw implicit feedback, pursing collaborative filtering approaches rather than *More Like This*. A simple extension to this work would be a hybrid approach, where a user-application matrix with binary entries (installed/not installed) could be factorized into latent factors, comparing the vectors extracted from the applications latent factor matrix between themselves for similarity. Finally, we can leverage the textual metadata and our fine-tuned model in an attempt to infer application quality, alongside with the global relevancy metrics. For instance, textual reviews can be considered when extracting relevancy scores (Siddiqui et al., 2021). Also, in an attempt to maintain user security and privacy, textual metadata (e.g., name and description) can be used to detect spam applications and/or applications that unwillingly collect user data (Seneviratne et al., 2017; Karunanayake et al., 2020; Hu et al., 2020; Rocha et al., 2020; Beg et al., 2021).

CRediT authorship contribution statement

João Coelho: Conceptualization, Methodology, Data curation, Software, Validation, Investigation, Writing – original draft, Writing – review & editing. **Diogo Mano:** Data curation, Software, Validation, Writing – review & editing. **Beatriz Paula:** Validation, Investigation, Writing – review & editing. **Carlos Coutinho:** Conceptualization, Supervision, Validation, Writing – original draft, Writing – review & editing, Project administration. **João Oliveira:** Conceptualization, Supervision, Validation, Writing – original draft, Writing – review & editing. **Ricardo Ribeiro:** Conceptualization, Supervision, Validation, Writing – original draft, Writing – review & editing. **Fernando Batista:** Conceptualization, Supervision, Validation, Writing – original draft, Writing – review & editing, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be published upon acceptance.

Acknowledgments

This work was supported by PT2020 project number 39703 (AppRecommender) and by national funds through FCT – Fundação para a Ciência e a Tecnologia, Portugal with reference UIDB/50021/2020.

References

AlRossais, N., Kudenko, D., Yuan, T., 2021. Improving cold-start recommendations using item-based stereotypes. User Model. User-Adapt. Interact. 31 (5).

Askari, B., Szlichta, J., Salehi-Abari, A., 2021. Variational autoencoders for top-k recommendation with implicit feedback. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '21, Association for Computing Machinery, New York, NY, USA, pp. 2061–2065. http://dx.doi.org/10.1145/3404835.3462986.

Beg, S., Anjum, A., Ahmad, M., Hussain, S., Ahmad, G., Khan, S., Choo, K.-K.R., 2021. A privacy-preserving protocol for continuous and dynamic data collection in IoT enabled mobile app recommendation system (MARS). J. Netw. Comput. Appl. 174.

Bhandari, U., Sugiyama, K., Datta, A., Jindal, R., 2013. Serendipitous recommendation for mobile apps using item-item similarity graph. In: Banchs, R.E., Silvestri, F., Liu, T.-Y., Zhang, M., Gao, S., Lang, J. (Eds.), Information Retrieval Technology. pp. 440–451.

Bialecki, A., Muir, R., Ingersoll, G., 2012. Apache Lucene 4. In: Proceedings of the SIGIR Workshop on Open Source Information Retrieval. pp. 17–24.

Chen, J., Cao, B., Liu, J., Li, B., 2020. MR-UI: A mobile application recommendation based on user interaction. In: 2020 IEEE International Conference on Web Services. ICWS, pp. 134–141. http://dx.doi.org/10.1109/ICWS49710.2020.00025.

Chen, N., Hoi, S.C., Li, S., Xiao, X., 2015. SimApp: A framework for detecting similar mobile applications by online kernel learning. In: Proceedings of the Eighth ACM International Conference on Web Search and Data Mining. WSDM '15, Association for Computing Machinery, New York, NY, USA, pp. 305–314. http://dx.doi.org/10.1145/2684822.2685305.

Coelho, J., Neto, A., Tavares, M., Coutinho, C., Oliveira, J., Ribeiro, R., Batista, F., 2021a. Transformer-based language models for semantic search and mobile applications retrieval. In: Cucchiara, R., Fred, A., Filipe, J. (Eds.), Proceedings of the 13th International Joint Conference on Knowledge Discovery and Information Retrieval, vol. 1. pp. 225–232. http://dx.doi.org/10.5220/0010657300003064, URL https://www.scitepress.org/PublicationsDetail.aspx?ID=SGs/9PKpzaQ=&t=1.

Coelho, J., Neto, A., Tavares, M., Coutinho, C., Ribeiro, R., Batista, F., 2021b. Semantic search of mobile applications using word embeddings. In: Queirós, R., Pinto, M., Simões, A., Portela, F., Pereira, M.J.a. (Eds.), 10th Symposium on Languages, Applications and Technologies, SLATE 2021. In: Open Access Series in Informatics (OASIcs), vol. 94, Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl, Germany, pp. 12:1–12:12. http://dx.doi.org/10.4230/OASIcs.SLATE.2021.12, URL https://drops.dagstuhl.de/opus/volltexte/2021/14429.

Colucci, L., Doshi, P., Lee, K., Liang, J., Lin, Y., Vashishtha, I., Zhang, J., Jude, A., 2016. Evaluating item-item similarity algorithms for movies. In: Proceedings of the Conference on Human Factors in Computing. pp. 2141–2147. http://dx.doi.org/10.1145/2851581.2892362.

Covington, P., Adams, J., Sargin, E., 2016. Deep neural networks for YouTube recommendations. In: Proceedings of the 10th ACM Conference on Recommender Systems. RecSys '16, Association for Computing Machinery, New York, NY, USA, pp. 191–198. http://dx.doi.org/10.1145/2959100.2959190.

Devlin, J., Chang, M.-W., Lee, K., Toutanova, K., 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). Association for Computational Linguistics, Minneapolis, Minnesota, pp. 4171–4186. http://dx.doi.org/10.18653/v1/N19-1423, URL https://aclanthology.org/N19-1423.

Gao, L., Callan, J., 2021. Unsupervised corpus aware language model pre-training for dense passage retrieval. ArXiv arXiv:2108.05540.

Hu, Y., Wang, H., He, R., Li, L., Tyson, G., Castro, I., Guo, Y., Wu, L., Xu, G., 2020. Mobile app squatting. In: Proceedings of the Web Conference 2020. WWW '20, Association for Computing Machinery, New York, NY, USA, pp. 1727–1738. http://dx.doi.org/10.1145/3366423.3380243.

Johnson, J., Douze, M., Jégou, H., 2017. Billion-scale similarity search with GPUs. ArXiv arXiv:1702.08734.

Karunanayake, N., Rajasegaran, J., Gunathillake, A., Seneviratne, S., Jourjon, G., 2020. A multi-modal neural embeddings approach for detecting mobile counterfeit apps: A case study on Google Play store. IEEE Trans. Mob. Comput. 21 (1).

Koren, Y., Bell, R.M., Volinsky, C., 2009. Matrix factorization techniques for recommender systems. Computer 42 (8).

Leng, H., Paulino, C., Haider, M., Lu, R., Zhou, Z., Mengshoel, O., Brodin, P.-E., Forgeat, J., Jude, A., 2018. Finding similar movies: Dataset, tools, and methods. In: Proceedings of the International Conferences in Central Europe on Human Computer Interaction. pp. 115–124. http://dx.doi.org/10.24132/CSRN.2018.2802.15.

Liang, D., Krishnan, R.G., Hoffman, M.D., Jebara, T., 2018. Variational autoencoders for collaborative filtering. In: Champin, P.-A., Gandon, F.L., Lalmas, M., Ipeirotis, P.G. (Eds.), WWW. ACM, pp. 689–698, URL http://dblp.uni-trier.de/db/conf/www/www2018.html#LiangKHJ18.

Lin, J., Nogueira, R., Yates, A., 2020. Pretrained transformers for text ranking: BERT and beyond. ArXiv arXiv:2010.06467.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V., 2019. RoBERTa: A robustly optimized BERT pretraining approach. ArXiv arXiv:1907.11692.

Lops, P., de Gemmis, M., Semeraro, G., 2011. Content-based recommender systems: State of the art and trends. In: Recommender Systems Handbook. Springer US, Boston, MA, pp. 73–105. http://dx.doi.org/10.1007/978-0-387-85820-3_3.

Loshchilov, I., Hutter, F., 2019. Decoupled weight decay regularization. In: 7th International Conference on Learning Representations, ICLR 2019, New Orleans, la, USA, May 6-9, 2019. OpenReview.net, URL https://openreview.net/forum?id=Bkg6RiCqY7.

van der Maaten, L., Hinton, G., 2008. Visualizing data using t-SNE. J. Mach. Learn. Res. 9 (86).

Messina, P., Dominguez, V., Parra, D., Trattner, C., Soto, A., 2019. Content-based artwork recommendation: integrating painting metadata with neural and manually-engineered visual features. User Model. User-Adapt. Interact. 29 (2).

Nogueira, R., Cho, K., 2019. Passage re-ranking with BERT. ArXiv arXiv:1901.04085.

Ouyang, Y., Guo, B., Tang, X., He, X., Xiong, J., Yu, Z., 2021. Mobile app cross-domain recommendation with multi-graph neural network. ACM Trans. Knowl. Discov. Data 15 (4).

Park, D.H., Liu, M., Zhai, C., Wang, H., 2015. Leveraging user reviews to improve accuracy for mobile app retrieval. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '15, Association for Computing Machinery, New York, NY, USA, pp. 533–542. http://dx.doi.org/10.1145/2766462.2767759.

Pereira, Í.M., Ferreira, A.A., 2019. An item-item similarity approach based on linked open data semantic relationship. In: Proceedings of the 25th Brazillian Symposium on Multimedia and the Web. WebMedia '19, Association for Computing Machinery, New York, NY, USA, pp. 425–432. http://dx.doi.org/10.1145/3323503.3349547.

Qin, Z., Zhuang, H., Jagerman, R., Qian, X., Hu, P., Chen, D.C., Wang, X., Bendersky, M., Najork, M., 2021. Bootstrapping recommendations at chrome web store. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining. KDD '21, Association for Computing Machinery, New York, NY, USA, pp. 3483–3491. http://dx.doi.org/10.1145/3447548.3467099.

Qu, Y., Ding, Y., Liu, J., Liu, K., Ren, R., Zhao, W.X., Dong, D., Wu, H., Wang, H., 2021. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, Online, pp. 5835–5847. http://dx.doi.org/10.18653/v1/2021.naacl-main.466, URL https://aclanthology.org/2021.naacl-main.466.

Ram, O., Shachaf, G., Levy, O., Berant, J., Globerson, A., 2021. Learning to retrieve passages without supervision. ArXiv arXiv:2112.07708.

Reimers, N., Gurevych, I., 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP. Association for Computational Linguistics, Hong Kong, China, pp. 3982–3992. http://dx.doi.org/10.18653/v1/D19-1410, URL https://aclanthology.org/D19-1410.

Robertson, S., Zaragoza, H., 2009. The probabilistic relevance framework: BM25 and beyond. Found. Trends Inf. Retr. 3 (4).

Rocha, T., Souto, E., El-Khatib, K., 2020. Functionality-based mobile application recommendation system with security and privacy awareness. Comput. Secur. 97.

Seneviratne, S., Seneviratne, A., Kaafar, M.A., Mahanti, A., Mohapatra, P., 2017. Spam mobile apps: Characteristics, detection, and in the wild analysis. ACM Trans. Web 11 (1).

Shenbin, I., Alekseev, A., Tutubalina, E., Malykh, V., Nikolenko, S.I., 2020. Recvae: A new variational autoencoder for top-N recommendations with implicit feedback. In: Proceedings of the 13th International Conference on Web Search and Data Mining. WSDM '20, Association for Computing Machinery, New York, NY, USA, pp. 528–536. http://dx.doi.org/10.1145/3336191.3371831.

Siddiqui, S., Faisal, M.S., Khurram, S., Irshad, A., Baz, M., Hamam, H., Iqbal, N., Shafiq, M., 2021. Quality prediction of wearable apps in the Google Play store. Intell. Autom. Soft Comput. 32.

Steck, H., 2019. Embarrassingly shallow autoencoders for sparse data. In: The World Wide Web Conference. WWW '19, Association for Computing Machinery, New York, NY, USA, pp. 3251–3257. http://dx.doi.org/10.1145/3308558.3313710.

Trattner, C., Jannach, D., 2020. Learning to recommend similar items from human judgments. User Model. User-Adapt. Interact. 30 (1).

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L.u., Polosukhin, I., 2017. Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc., pp. 5998–6008, URL https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wang, C., Agrawal, A., Li, X., Makkad, T., Veljee, E., Mengshoel, O., Jude, A., 2017. Content-based top-N recommendations with perceived similarity. In: 2017 IEEE International Conference on Systems, Man, and Cybernetics. SMC, pp. 1052–1057. http://dx.doi.org/10.1109/SMC.2017.8122750.

Wu, C., Wu, F., Qi, T., Huang, Y., 2021. Empowering news recommendation with pre-trained language models. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. SIGIR '21, Association for Computing Machinery, New York, NY, USA, pp. 1652–1656. http://dx.doi.org/10.1145/3404835.3463069.

Xie, F., Chen, L., Ye, Y., Liu, Y., Zheng, Z., Lin, X., 2018. A weighted meta-graph based approach for mobile application recommendation on heterogeneous information networks. In: Proceedings of the International Conference on Service-Oriented Computing. pp. 404–420.

Yagci, A.M., Aytekin, T., Gürgen, F.S., 2017. Scalable and adaptive collaborative filtering by mining frequent item co-occurrences in a user feedback stream. Eng. Appl. Artif. Intell. 58.

Zaragoza, H., Craswell, N., Taylor, M., Saria, S., Robertson, S., 2004. Microsoft cambridge at TREC-13: Web and HARD tracks. In: Proceedings of TREC 2004. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.61.965.

Zhang, X., Qin, J., Zheng, J., 2021. A social recommendation based on metric learning and users' co-occurrence pattern. Symmetry 13 (11).