Multigraph Approach Towards a Scalable, Robust look-alike Audience Extension System

Ernest Kirubakaran Selvaraj, Tushar Agarwal, Nilamadhaba Mohapatra, Swapnasarit Sahu
Zeotap India Pvt Ltd
Bangalore, India
{ernest.kirubakaran,tushar.agarwal,nilamadhaba.mohapatra,swapnasarit.sahu}@zeotap.com

ABSTRACT

In online advertising, finding the right audience is critical for the success of a campaign. One common way of finding the right audience is to find users with traits similar to the users who have responded positively to the campaign in the past. The small pool of users who have responded positively to the campaign is known as the seed set and the goal here is to reach a bigger audience with traits very similar to that of the seed set. This technique, popularly known as look-alike audience extension, gets increasingly challenging with the scale and high sparsity of data commonly encountered in the advertising domain. In this paper, we present a novel multigraph-based audience extension and scoring system, which works well with high-dimensional sparse data and can be scaled easily to millions of users. Our experimental results on large real-world data demonstrate significant improvement in the performance of our approach over the existing architectures.

CCS CONCEPTS

• **Human-centered computing** → **User models**; • **Information systems** → *Online advertising*; • **Theory of computation** → Computational advertising theory.

KEYWORDS

online advertising, lookalike modeling, user representation learning

ACM Reference Format:
Ernest Kirubakaran Selvaraj, Tushar Agarwal, Nilamadhaba Mohapatra, Swapnasarit Sahu. 2021. Multigraph Approach Towards a Scalable, Robust look-alike Audience Extension System. In *AdKDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, August, 2021, Singapore.* ACM, New York, NY, USA, 6 pages.

1 INTRODUCTION

Targeted advertising is a form of online advertising where the advertiser targets a specific set of online users with particular traits and behavior. The goal of the advertiser is to reach maximum users at the same time to maximize the return on ad spending. Hence the advertisers must target the right set of users to improve ad clicks and conversion. One way of finding the right set of large online audiences is known as look-alike modeling. In look-alike modeling, the goal is to find a larger audience that is very much similar to a smaller set of audiences who have positively responded to the ad, for example, by clicking on the ad. The smaller set of audiences is known as the seed set. The similarity between the seed set and the larger audience is measured in terms of the similarities in demographic attributes, interests & intents, product purchase history, and online behavior, among others. Finding similar users is a particularly challenging problem because the pool for the extended target audience is usually extremely big. While this large data leads to computational issues, the nature of the online advertising business demands the look-alike system to have extremely low latency.

Another challenge look-alike systems face is the nature of data in the advertising domain. The data is high-dimensional and extremely sparse with a lot of missing values. Some of the data is static like demographic details, while the majority of the data such as user intent, interests, mobile apps usage, etc is highly dynamic and temporal. Demographic data available to the advertiser includes age, gender, location of the users along with other attributes like financial information of the users.

In this paper, we propose a novel multigraph-based look-alike system that is highly scalable and shows improved performance over existing look-alike systems. Our contributions are summarized as follows:

- We present a weighted multigraph-based look-alike system, where user-to-user similarity is captured in the form of a weighted multigraph.
- We develop a scoring method to select users from amongst the multigraph neighbors of the seed set for inclusion in the look-alike extension.
- We provide empirical evidence on the improved performance of the proposed model when compared to other models in real-world experiments. We have also benchmarked our model with the Adform click prediction dataset[15].

2 RELATED WORKS

Various look-alike models have been proposed in the past, including rule-based models [10, 14], classification models [12, 13], similarity models [5, 8, 9], and deep learning models [2, 4, 6]. [10] use a rule-based associative classifier for tail campaigns with very small conversion rates. [14] use a model based on the Bayes rule, built on hand-crafted user segments. Though rule-based models are easy to implement, they are too simplistic to capture complex user behavior. Classification models work by building a model to separate the seed set users from a negative set of users. Various strategies are

AdKDD '21, August, 2021, Singapore

ACM ISBN 978-1-xxxx-xxxx-x/21/08…$15.00

Code: https://github.com/ernest-s/Multigraph-Lookalike

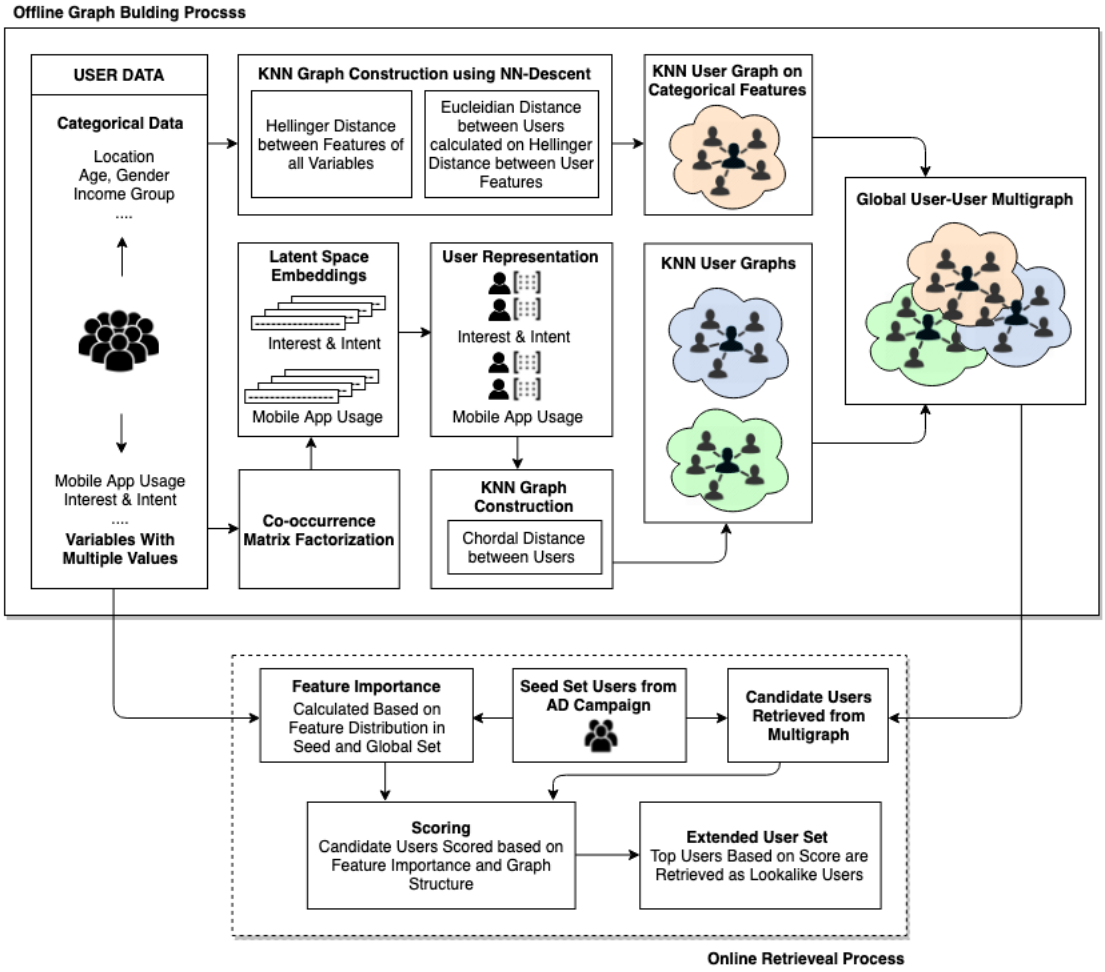

Figure 1: Multigraph Look-alike System Architecture and Pipeline

employed to get the negative set of users including sampling from users who were shown the ad but haven't clicked on it. [12] use such a model. [13] use k-means clustering to group user segments into different clusters and expand the segments by computing the distance from the candidate users to the cluster centroids. Similarity models work on the premise that users with similar profiles tend to behave similarly to ad campaigns. The model proposed by [5] use user similarity and the model proposed by [8, 9] employ locality-sensitive hashing (LSH) [16] technique to find a user similarity. The candidate users are picked based on their similarity to the seed set users, and a scoring methodology is employed to filter the candidate users. While LSH is a faster way to build user similarity, the computational cost of LSH based methods goes high when we want accurate results from the model [17]. Methods proposed by [2] build a user embedding, find seed representation by applying LSH on top of user embedding and score each user based on the seed representation. Then, [6] use another look-alike learning that uses an attention mechanism to predict look-alike similarity on top of a user embedding. The model proposed by [4] uses a multilayer perceptron model, trained to distinguish between seed set users and a negative set of users sampled using a spy sampling technique. More recently, two-stage hybrid models[21][7] have been proposed where a user representation is learned using graph neural networks and, feedbacks from campaigns are used in the expansion stage.

3 THE PROPOSED LOOK-ALIKE MODEL

In this section, we formalize the problem and discuss our proposed multigraph algorithm.

3.1 Problem Definition

Given a global set of users U with features from a set F, we receive a set of seed set users S from the advertiser. Seed set users are the users who have responded positively (either by clicking on the ad or completing a purchase) to an ad campaign. Here $|S| << |U|$. The goal is then to find users from the global set who are similar to

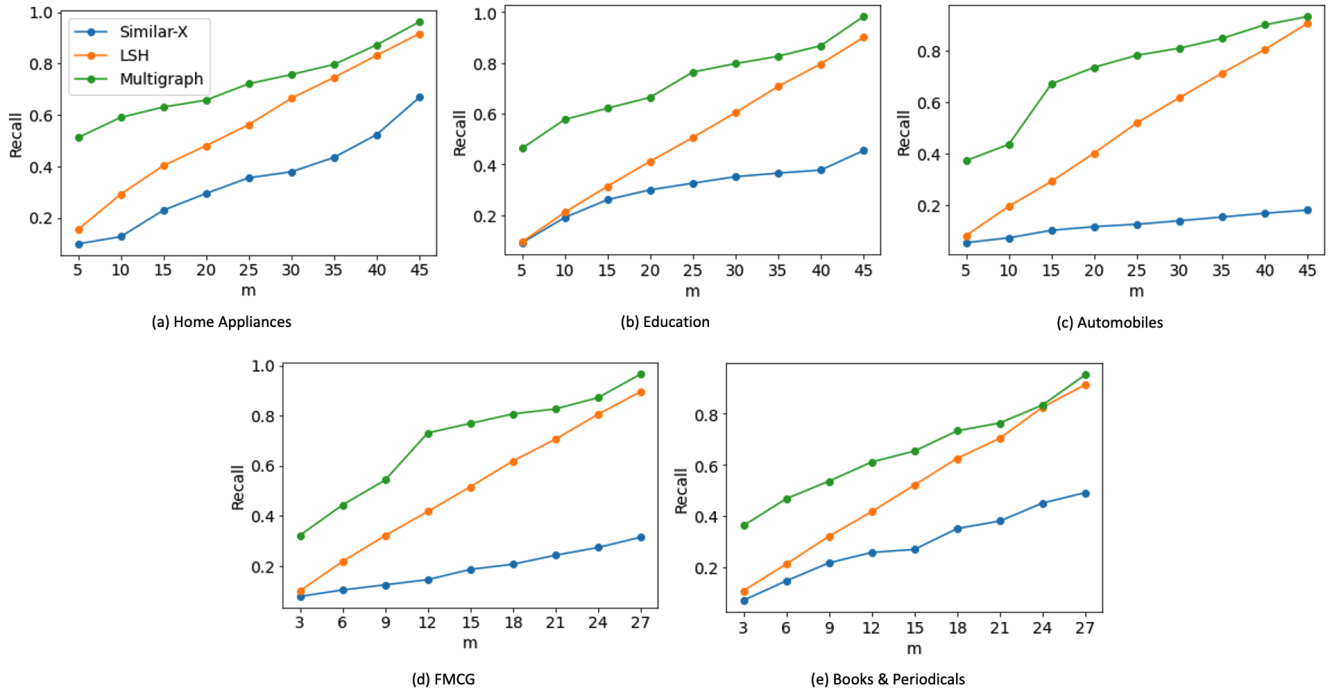

Figure 2: Recall at different values of m for the 5 campaigns

seed set users. The advertiser can then target similar users. The key assumption here is that users with similar features tend to behave similarly (by responding positively to the ad campaign).

3.2 Multigraph Approach

Before the user-user graph is built, the user features are grouped into various categories such as demographics, app usage data, interest & intent data, and product purchase data. Certain sets of features such as demographics can have only one value per feature whereas other features such as product purchase data can have multiple products per user. For every category of data, a user-user knn-graph is built. In the graph, every user is connected to k most similar users. The graphs are weighted and the edge weight is the similarity between the users it connects. To build the knn-graph using variables with high cardinality, we first learn a latent space embedding for those variables. And the knn-graph is built on the latent space embedding. The graph building process is an offline one-time process.

The global user-user graph helps in finding the right set of candidates for look-alike extension in near real-time. During retrieval, a set of seed users are received. These users are the ones who have responded positively to an ad campaign. The seed set is then matched with the global user-user multigraph and for every user in the seed set, the neighbors are retrieved as candidate users for look-alike extension. Then every candidate user is ranked based on the closeness to the seed set by a scoring algorithm. Finally, the top users from the candidate user set based on the score are provided as look-alike extension. Figure 1 shows the overall architecture of our model. Grouping variables into various categories and building separate graphs for each category ensure that each category gets equal importance and no single category dominates the model. In the multigraph, all the graphs are equally weighted.

3.3 Multigraph Construction

The data available on users is grouped into multiple categories such as demographic data, interest & intent data, apps usage data, product purchase data, etc., and a weighted multigraph is constructed on these grouped variables. Every category of data will have its own edge type. And the edges will have weights depicting how strong the relationship between the two nodes it is connecting. Two users who are neighbors in one group need not be neighbors in others. For example, two users may have similar demographics but their interest & intent may have nothing in common. In this case, they will have a demographics edge between them but not an interest & intent edge. For some campaigns, demographics may play an important role and for others, demographics may not be important at all. Building a multigraph based on grouped variables helps in deciding which group of variables is important in the given seed set. A K-Nearest Neighbor Graph(K-NNG) is built for every category of data available namely demographics, app usage, location, interest & intent, etc. and the graphs are combined to form a multigraph.

3.3.1 NN-Descent. The goal of K-NNG is to build a graph where every node is connected to its closest K nodes. K-NNG has an algorithmic complexity of $O(n^2)$ complexity and is not suitable for large datasets. *NN-Descent*[3] method overcomes this problem by employing an iterative approach to build the K-NNG while keeping the complexity to an empirical level of $O(n^{1.14})$. This method works by starting with a random graph and iteratively updating the neighbors of every node by exploring its neighbors' neighbors. The NN-Descent method produces high accurate K-NNG graphs

and it works for any underlying distance function. Moreover, the algorithm is guaranteed to converge if appropriate K is chosen. The K chosen has to be greater than c^3 where c is the *growing constant* of the input space. *Growing constant* of a growth-restricted metric space V is defined as a constant c s.t.

$$|B_{2r}(v)| \leq c|B_r(v)|, \qquad \forall v \in V \tag{1}$$

where, $|B_r(v)|$ is the number of neighbors of v within distance r from v in the metric space V.

3.3.2 Distance Metric for Categorical Variables. An appropriate distance metric has to be defined to build the graphs. For demographic data, if we consider each of the features such as age, gender, and location as a dimension, to find the Euclidean distance between any two users in this n-dimensional space, we first need to calculate how far any two categories are apart in a given feature dimension. For example, if we consider two users having location feature as city A and city B among other features, to calculate the user distance in the n-dimensional feature space, we need to find the relative distance in the location dimension between cities A and B. We use the Hellinger Distance to find the relative distance between any two feature values. Hellinger Distance between two feature values j and k belonging to feature f is given by

$$HD(f_j, f_k) = \sqrt{1 - \sum_{\substack{l \in f' \\ \forall f' \in F, f' \neq f}} \sqrt{p_{sj}(f' = f'_l) p_{sk}(f' = f'_l)}} \tag{2}$$

where F is the feature set in the global user set U, l is the feature value, $p_{sj}(f' = f'_l)$ is the probability that the feature f' takes value l in the subset sj and the subset sj is the subset of the global user set U where the feature f takes value j. In the above example, cities A and B will have a lower score if the feature distribution of users in both the cities is similar and a higher score if the feature distribution is dissimilar. Once we have the relative distance between feature values, the Euclidean Distance between any two users a and b can be calculated using

$$d(a, b) = \sqrt{(\sum_{f \in F} HD(f_a, f_b))^2} \tag{3}$$

where f_a is the value user a takes for feature f.

3.3.3 Embeddings for High Cardinality Features. In addition to the demographics, we also have other features such as app usage and interest and intent which are high cardinal in nature. These features can also take multiple values per user. When the high cardinal data is used in a one-hot encoded form, all data points are perpendicular to each other and the similarity between target values is not captured. Thus, there is no inherent distance metric between target values unless they are pushed into a latent space. Also, *NN-Descent* method won't work in high dimensions because of the phenomenon of *hubness*[1]. *Hubness* refers to the tendency of high-dimensional data to reside in hubs inside the space rather than uniformly distributed across space. And this phenomenon will cause the *NN-Descent* algorithm to converge to local minima rather than global minima.

To overcome these two issues, we first build a latent representation of high cardinal columns using *GloVe*[11] architecture. *GloVe* architecture is used to get word embeddings by factorizing the word-word co-occurrence matrix from a large corpus. The co-occurrence matrix is constructed by counting the words occurring together in a context window. In our model, the users provide the context window. For example, if a particular user may have used apps a, b and c, then all these three apps are in the same context. The global co-occurrence matrix is constructed for all users. The latent space embedding for every target value in the feature is obtained by factorizing the global co-occurrence matrix. The glove vectors are trained my minimizing the function J, given by

$$J = \sum_{i,j=1}^{V} f(X_{ij})(\omega_i^T \omega_j + b_j + b_j - log(X_{ij}))^2 \tag{4}$$

where V is the number of values that feature f can take, X_{ij} is the co-occurrence matrix, $f(X_{ij})$ is a weight function to handle noisy data, ω_i is the embedding vector of the i th value of feature f and, b_i is the bias term. The latent space dimension k « *input feature dimension*. By factorizing the global co-occurrence matrix, a dense representation for each target value in a much smaller space is obtained. The dense representation also captures the relationships between target values. Thus, every user who has used n unique apps in the past is represented by a k x n matrix where the columns of the matrix represent the apps in the latent space.

3.3.4 Modified Chordal Distance. To build a weighted user-user graph for high cardinal features like app usage, we need to define a distance metric. For these features, every user is represented as a k x n matrix where k is the latent space dimension of the feature and n is the number of target values the user has for that feature. Here, every user can have a different number of target values. For example, one user might have apps usage data for 5 apps and another user might have app usage data for 11 apps. Thus, user matrices are not of the same dimension. Every user can be thought of occupying an n-dimensional subspace inside the k-dimensional space. Different methods have been used in the past to find the distance between two subspaces. Wolf and Shashua[20] used principal angles between subspaces. The similarity measure they employed is $\prod_{i=1}^{k} cos(\theta_i)^2$, where θ's are the principal angles between the two subspaces. However, this method requires the two subspaces to be equidimensional. [19] used a modified version of Chordal distance to measure the distance between two subspaces. In their approach, the distance between two subspaces is given by

$$d(U, V) = \sqrt{max(m, n) - \sum_{i=1}^{m} \sum_{j=1}^{n} (u_i^T v_j)^2} \tag{5}$$

where U and V are m-dimensional and n-dimensional subspaces in $\mathbb{R}^k$ respectively and u_1,u_2,...,u_m and v_1,v_2,...,v_n are orthonormal bases of U and V, respectively. The above distance metric is a complete metric [19], [18] and can be used in *NN-Descent* as a distance metric. We use the above metric to build a user-user graph for high cardinal features that can have multiple values per user.

3.4 Scoring Methodology

When a seed set is received from a campaign, a set of potential candidate look-alike users can be formed by selecting the neighbors of the seed set users in the global multigraph. Then we need to

score the candidate users based on their likelihood that they belong to the seed set. Information Value captures the importance of a user feature by comparing the feature distributions in the seed set and the global user set. If the distribution is different, then the feature is important and vice versa.

$$IV(f \in F) = \sum_{j \in f} (p_u - p_s)\, ln\left(\frac{p_u + \epsilon}{p_s + \epsilon}\right) \tag{6}$$

$$p_u(j) = \frac{\sum_U \mathbb{1}(f = j)}{|U|}, \quad p_s(j) = \frac{\sum_S \mathbb{1}(f = j)}{|S|}$$

where, U is the global set of users, S is the seed set users, F is the set of features a user can have, j is a value feature f can take, and f=j for a user u indicates the feature f has a value j for the user u. While IV captures the feature importance, at the user level, the IV needs to be weighted based on the value j the user takes for the variable f. The value $p_s(k)$, which captures the probability of j in the seed set S. The weighted IV score for a user is given below. Here, $x_{u,f=j} \in \{0, 1\}$ is an indicator function which is *1* if the feature f for the user u is j, 0 otherwise and C is the set of all candidate users.

$$Weighted\, IV(c \in C) = \sum_{f \in F} \sum_{j \in f} p_s(k)\, x_{c,f=j}\, IV(f) \tag{7}$$

Apart from information value, the structure of the graph itself can give some information on the importance of a candidate user. For every candidate user, the number of seed set users in its neighborhood is also an indicator of the importance of candidate users. Similarly, the number of edges between the candidate users and the seed set users indicating similarity across different categories of data is also an indicator of importance. The formula for the score of a candidate user c, using these three values is given below. Here, N_c is the number of seed set users in the neighborhood of c in the global user-user graph, E_c is the number of edges between c and its closest seed set user.

$$Score(c \in C) = N_c\, E_c \sum_{f \in F} \sum_{j \in f} p_s(k)\, x_{c,f=k}\, IV(f) \tag{8}$$

Users in the candidate set are scored and the top users are filtered as look-alike users.

4 EXPERIMENTAL EVALUATION

In this section, we discuss the two methods we use to evaluate the multigraph look-alike model.

4.1 Online AB Testing

In the first set of experiments, we ran a series of advertisement campaigns and compared the performance of our model and the LSH[9] and Similar-X[5] models.

4.1.1 Dataset. The dataset consists of approximately 375 million user profiles. To test the model, 3 sets of user profile information were used; demographics (age, gender, income group, parental status, etc.), interest & intent data and app usage data (installed mobile apps). The demographic data is categorical whereas the other two can have multiple values per user. For the demographics, Hellinger Distance followed by Euclidean Distance was used as a metric to build the subgraph. For interest & intent and app usage data, embeddings are learned and the graphs are built on embeddings using modified Chordal distance.

Campaign	Baseline	Similar-X	LSH	Multigraph
Appliances	1.85%	4.05%	7.73%	**8.02%**
Education	1.25%	3.21%	7.07%	**7.38%**
Automobiles	1.36%	2.67%	7.77%	**8.67%**
FMCG	1.68%	2.20%	3.10%	**3.25%**
Books	2.21%	2.45%	4.15%	**5.38%**

Table 1: CTR Rates for various campaigns

4.1.2 Campaign Performance. We ran a set of 5 online advertising campaigns covering a diverse set of industries. The metric used in these experiments is the click-through rate (CTR) defined as the percentage of users who clicked on an ad (clicks) to the total number of users who were shown the ad (impressions).

$$CTR = \frac{clicks}{impressions} \tag{9}$$

For every campaign in the experiment, the initial run happens on segments handcrafted by the advertiser. The CTR value from the initial run is considered as the baseline value. From the clicks, we get the seed set users. The seed set is used by our multigraph model to get an extended set of users. The second phase of the campaign is run on an equal number of extended set of users from all three models and the CTR rate is recorded. Results show that our multigraph model outperforms both models LSH and similar-X models. Table 1 shows the CTR rates during the experiment for various models. Here our multigraph model shows better performance than single graph models. Performance in some campaigns shows a wider gap than the rest because user similarity is a strong factor for clicks for some of the campaigns. The LSH model uses MinHash and the similar-X model uses Arcos algorithm to find neighbors. The metrics used in our model is explained in section 3. Table 2 shows the overall percentage improvement for all the 3 models.

Model	% Increase in CTR
Similar-X	62
LSH Model	233
Multigraph Model	263

Table 2: Percentage CTR improvement over advertiser generated segment

4.1.3 Recall Experiment. In addition to the online A/B test, we ran another experiment to evaluate how far we can recover the original seed set by extending it on a subset of the seed set. The steps involved in the experiment are as follows:

(1) Seed set S, consisting of users who have clicked on the ad is received from an ongoing campaign
(2) Seed set is randomly split into two equal sets S_1 and S_2
(3) The set S_1 is extended and we assess how many users from set S_2 are present in the extension X_e using the metric:

$$recall_{@m} = \frac{|X_e \cap S_2|}{|S_2|}$$

The recall rate is calculated at different values of m for multiple seed sets. In figure 2, we can see that the multigraph model outperforms the other two models in the recovery task..

4.2 Seed Set Recovery

We ran another set of experiments on Adform data to evaluate how far we can recover the original seed set by extending on a subset of the seed set. The dataset consists of impressions served by Adform for a particular campaign and a select set of features. The features are hashed and couple of them can take multiple values. The dataset also has a binary feature indicating whether the ad was clicked or not. The steps involved in the experiment are as follows:

(1) A multigraph was built on the dataset with 3 subgraphs, one for the categorical columns and the rest for the two columns that have multiple values per row.
(2) A set of users who had clicked the ad as C from the dataset are extracted and a random subset S, consisting of 50,000 users from C was taken as Seed Set.
(3) The set S is extended and we assess how many users from extended set X had clicked on the ad using the metric:

$$precision@k = \frac{|X \cap C|}{|X|}$$

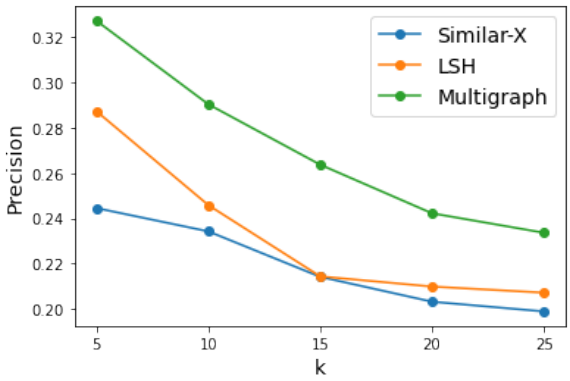

Figure 3: Precision@k for different k values

The multigraph model shows higher precision rates in figure 3.

5 CONCLUSION

In this paper, we present a multigraph look-alike audience extension system. The core approach for our model is based on a multigraph built on different categories of user data. Allowing different graphs for different categories makes neighborhood searches more robust and accurate. The model is computationally efficient during prediction time as prediction involves only the retrieval of candidate users and scoring them. The model also scales well to millions of data points. Real-world experiments show that our model achieves better CTR rates than existing audience extension models. Our model also gives superior performance in segment retrieval tasks. Our approach provides an accurate extension set as evidenced by the experimental results due to the balanced importance given to different sets of user features in a multigraph and a more accurate graph built using deterministic methods. In future work, we plan to extend the method to accommodate changes in user intent signals over time, as user intent is typically short-lived and changes often.

REFERENCES

[1] Brankica Bratić, Michael E Houle, Vladimir Kurbalija, Vincent Oria, and Miloš Radovanović. 2018. Nn-descent on high-dimensional data. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*. 1–8.
[2] Stephanie deWet and Jiafan Ou. 2019. Finding Users Who Act Alike: Transfer Learning for Expanding Advertiser Audiences. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2251–2259.
[3] Wei Dong, Charikar Moses, and Kai Li. 2011. Efficient k-nearest neighbor graph construction for generic similarity measures. In *Proceedings of the 20th international conference on World wide web*. 577–586.
[4] Jinling Jiang, Xiaoming Lin, Junjie Yao, and Hua Lu. 2019. Comprehensive Audience Expansion based on End-to-End Neural Prediction. In *Proceedings of the SIGIR 2019 Workshop on eCommerce, co-located with the 42st International ACM SIGIR Conference on Research and Development in Information Retrieval, eCom@ SIGIR 2019, Paris, France, July 25, 2019*. CEUR-WS. org.
[5] Haishan Liu, David Pardoe, Kun Liu, Manoj Thakur, Frank Cao, and Chongzhe Li. 2016. Audience expansion for online social network advertising. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 165–174.
[6] Yudan Liu, Kaikai Ge, Xu Zhang, and Leyu Lin. 2019. Real-time Attention Based Look-alike Model for Recommender System. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2765–2773.
[7] Zhining Liu, Xiao-Fan Niu, Chenyi Zhuang, Yize Tan, Yixiang Mu, Jinjie Gu, and Guannan Zhang. 2020. Two-Stage Audience Expansion for Financial Targeting in Marketing. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 2629–2636.
[8] Qiang Ma, Eeshan Wagh, Jiayi Wen, Zhen Xia, Robert Ormandi, and Datong Chen. 2016. Score Look-Alike Audiences. In *2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW)*. IEEE, 647–654.
[9] Qiang Ma, Musen Wen, Zhen Xia, and Datong Chen. 2016. A sub-linear massive-scale look-alike audience extension system. In *Proceedings of the 5th International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*.
[10] Ashish Mangalampalli, Adwait Ratnaparkhi, Andrew O Hatch, Abraham Bagherjeiran, Rajesh Parekh, and Vikram Pudi. 2011. A feature-pair-based associative classification approach to look-alike modeling for conversion-oriented user-targeting in tail campaigns. In *Proceedings of the 20th international conference companion on World wide web*. 85–86.
[11] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. 1532–1543.
[12] Yan Qu, Jing Wang, Yang Sun, and Hans Marius Holtan. 2014. Systems and methods for generating expanded user segments. US Patent 8,655,695.
[13] Archana Ramesh, Ankur Teredesai, Ashish Bindra, Sreenivasulu Pokuri, and Krishna Uppala. 2013. Audience segment expansion using distributed in-database k-means clustering. In *Proceedings of the Seventh International Workshop on Data Mining for Online Advertising*. 1–9.
[14] Jianqiang Shen, Sahin Cem Geyik, and Ali Dasdan. 2015. Effective audience extension in online advertising. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2099–2108.
[15] Enno Shioji. 2017. Adform click prediction dataset. https://doi.org/10.7910/DVN/TADBY7
[16] Malcolm Slaney and Michael Casey. 2008. Locality-sensitive hashing for finding nearest neighbors [lecture notes]. *IEEE Signal processing magazine* 25, 2 (2008), 128–131.
[17] Malcolm Slaney, Yury Lifshits, and Junfeng He. 2012. Optimal parameters for locality-sensitive hashing. *Proc. IEEE* 100, 9 (2012), 2604–2623.
[18] Xichen Sun, Liwei Wang, and Jufu Feng. 2007. Further results on the subspace distance. *Pattern recognition* 40, 1 (2007), 328–329.
[19] Liwei Wang, Xiao Wang, and Jufu Feng. 2006. Subspace distance analysis with application to adaptive Bayesian algorithm for face recognition. *Pattern recognition* 39, 3 (2006), 456–464.
[20] L Wof and Amnon Shashua. 2003. Kernel principal angles for classification machines with applications to image sequence interpretation. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, Vol. 1. IEEE, I–I.
[21] Chenyi Zhuang, Ziqi Liu, Zhiqiang Zhang, Yize Tan, Zhengwei Wu, Zhining Liu, Jianping Wei, Jinjie Gu, Guannan Zhang, Jun Zhou, et al. 2020. Hubble: an Industrial System for Audience Expansion in Mobile Marketing. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2455–2463.