Soft Computing (2019) 23:11793–11799
https://doi.org/10.1007/s00500-018-03730-5

METHODOLOGIES AND APPLICATION

C-3PO: *C*lick-sequence-aware dee*P* neural network (DNN)-based *P*op-u*P*s rec*O*mmendation

I know you'll click

TonTon Hsien-De Huang[1,2] · **Hung-Yu Kao**[2]

Published online: 4 January 2019

Abstract
With the emergence of mobile and wearable devices, push notification becomes a powerful tool to connect and maintain the relationship with app users, but sending inappropriate or too many messages at the wrong time may result in the app being removed by the users. In order to maintain the retention rate and the delivery rate of advertisement, we adopt deep neural network (DNN) to develop a pop-up recommendation system "*C*lick-sequence-aware dee*P* neural network (DNN)-based *P*op-u*P*s rec*O*mmendation (C-3PO)" enabled by collaborative filtering-based hybrid user behavioral analysis. We further verified the system with real data collected from the product security master, clean master, and CM browser, supported by Leopard Mobile Inc. (Cheetah Mobile Taiwan Agency). In this way, we can know precisely about users' preference and frequency to click on the push notification/pop-ups, decrease the troublesome to users efficiently, and meanwhile increase the click-through rate of push notifications/pop-ups.

Keywords Click sequence aware · Deep learning · Deep neural network

1 Introduction

Smartphone has become a necessity in our life. According to the report in 2016 by International Data Corporation (IDC) 2016, the market share of Android smartphone in Q3 2016 has grown to 86.8%, becoming the operating system used by most majorities. On the other hand, according to the statistics by the software company App Annie, in Q3 2017, the number of downloads globally has reached 26 billion, not including app update or repeated downloads. The expenditure also broke the historical record. Users spent about 325 billion hours in total, increasing about 40% comparing to 2016. There is no relief on the growth of app economy. This brought the driving force to online advertising industry. In this industry, targeted advertising and its click-through rate forecast play an important role on overall user experience and revenue of a system. Meanwhile, apps usually send pop-ups to users as push notification service to notify user important information or alerts. In order not to make troubles to users, current method is to show simple graph and messages/texts on operation screen status column until the user drags down to show full contents. To open the message, just simply click on the column as shown in Fig. 1. Through reminding users their smartphone operating condition, we can increase the open rate and the number of advertisement display. Using push notification service properly can increase the using rate of apps and further increase the number of advertisement display. However, if push notification is used in an inappropriate way, it will become a troublesome to users, resulting the app to be removed. Before thinking about how to increase the efficiency of targeted advertising and its click-through rate, we should think about how to decrease the troublesome to users. It is important to forecast users' preference and frequency on push notifications precisely.

Communicated by V. Loia.

✉ TonTon Hsien-De Huang
TonTon@TWMAN.ORG

[1] Leopard Mobile Inc. (Cheetah Mobile Taiwan Agency), Taipei, Taiwan

[2] Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan, Taiwan

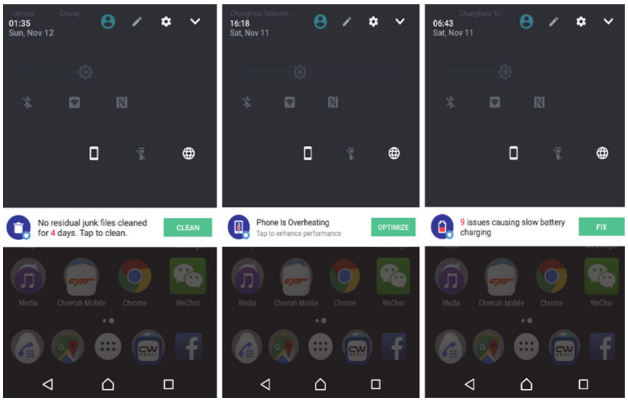

Fig. 1 Pop-ups example

In recent years, deep learning has made important progress in many fields including image recognition, speech recognition, and natural language processing. It is a prime time for the development of deep learning. Its essence is to carry on deep abstract excavation to the data characteristic through big data, in order to learn the effective characteristic expression and the complex mapping mechanism to establish effective data model. In the era of big data, the individual needs of users are constantly increasing. Facing a big data, one of the major research challenges is how to help users effectively obtain the information they need. At present, the most common type of information processing system is a search engine. Users submit queries and the system returns search results. The other is that users do not need to explicitly submit any inquiries and interest preferences. The system uses automated algorithms to push information. The classic one is a recommendation system.

Meanwhile, online advertising has been a hot topic and up until now there has been a number research papers and patents. Many start-up companies have also involved in this area and generated certain amount of revenue, and their forecasting techniques improve overtime. The mobility and usability of smartphone that shows information, such as personal data, browsing history, shopping history, financial details, and so on, allow tracking on user online behavior to build personalized knowledge to provide advertisement based on their preference, interest, and needs. The method is known as collaborative filtering. Then, we can make behavioral model of users. Users with similar profiles tend to behave similarly to advertisements. According to the data collected by our system, the behavior to push notification is highly alike to the behavior to online advertising. Considering the similarity between different users, we hope to increase the display rate of advertisement based on this observatory result. However, according to what we have known, seldom research has been done in this specific topic, especially for push notification service and targeted advertising and forecasting. The reason is that the revenue generated by push notifications is far less than the direct revenue from online advertising.

Therefore, the main contribution of this work is to verify our propose "***C***lick-sequence-aware dee***P*** neural network (DNN)-based ***P***op-u***P***s rec***O***mmendation (C-3PO)" that can prove the effect on decreasing troublesome of pop-ups and increasing the display and click-through rate of advertisement.

2 Related work

2.1 Deep learning

Recent success in deep learning research and development attracts people's attention. AlphaGo from Google DeepMind gains a huge success in Computer Go. The deep learning behind the AlphaGo receives huge attentions from both publics and academics (Silver et al. 2016). In 2015, Google released Tensorflow (Abadi et al. 2016) which provided a flexible framework for experimenting all kinds of deep neural network framework in mass-distributed training. Deep learning is a specific type of machine learning. More specifically, deep learning is an artificial neural network, in which multiple layers of neurons are interconnected with different weights and activation functions to learn the hidden relationship between input and output. Intuitively, input data are fed to the first layer that generates different combinations of the input (LeCun et al. 2015). These combinations, after the activation function, are fed to the second layer, and so on. Under the above procedures, different combinations of the outputs from previous layer can be seen as different representation of features. The weights on links between layers are adjusted according to backward propagations, depending on the distance or less function between true output label and the label calculated by neural network. Note that deep learning can be seen as a neural network with a large number of layers. After the above learning process via multiple layers, we can derive a better understanding and representation of distinguishable features, enhancing the detection accuracy (Goodfellow et al. 2016). Also, notice that the effectiveness of deep learning increases with the network size.

In addition to deep neural networks, the most well-known deep networks are convolutional neural networks (CNN). The representation of CNN includes AlexNet, VGG, GoogleNet, Inception-v3, and ResNet (Krizhevsky et al. 2012; Simonyan and Zisserman 2015; Szegedy et al. 2016; He et al. 2016). More specifically, CNN is composed of hidden layers, fully connected layers, convolution layers, and pooling layers. The hidden layers are used to increase the complexity of the model. If the same number of neural is associated with the input image, the number of parameters

can be significantly reduced, adapting to the function structure much properly.

2.2 Recommendation system

The popular topic recently is to profile user's behavior based on various historical records and click through to make forecast and improve the system. One method is to improve user experience on recommendation system (e.g., E-commerce) from voting or any satisfaction voting to leave specific explicit feedback, and another method is to store user browsing activities to actively follow user behavior as implicit feedback. Collaborative filtering (CF) is an implicit feedback which identifies new users' relation and forecast based on the relationship between users and inter-dependent relationship between items and recommend directly based on user purchasing behavior when user shows enough purchasing behavior (Pan et al. 2008). Koren et al. (2009) considered that the method based on CF is better than the method based on contents. Collaborative filtering problems are alike on social networking sites, tags for photographs, Web sites visited during a surfing session, articles bought by a customer, etc. Verstrepen and Goethals (2014) proposed user-based and item-based nearest neighbors algorithms: one-class collaborative filtering, and use this reformulation to propose a novel algorithm that incorporates the best of both worlds and outperforms state-of-the-art algorithms. Fabio Aiolli focus on memory-based collaborative filtering algorithms similar to the well-known neighbor-based technique for explicit feedback. Then, starting from the definition of suitable similarity and scoring functions and suggestions on how to aggregate multiple ranking strategies, the overall recommendation is defined (Aiolli 2013).

A representative work of using neural network models earlier can be traced back to Salakhutdinov et al. using a restricted Boltzmann machine for scoring prediction (Salakhutdinov et al. 2007). However, a major problem with this approach is that the weighting parameters connecting the hidden layer and the scoring layer are too large (for large data sets). Therefore, the author further proposes to use W to decompose W into two low-rank matrices to reduce the parameter size. Zheng et al. (2016) proposed to use the neural autoregressive distribution estimator to improve the above problems. Experiments show that the proposed method can achieve very good results. Wu et al. (2016) used the denoising auto-encoder to perform top-N item recommendation. The input is the preference for the item added to the noise, and the output is the user's original rating for the item. The item predictions are made through learning nonlinear mapping relationships.

The same thing in these jobs is that they are all based on the user's original score (or feedback) to dig deep data pattern features. There are still some attempting to use the deep neural network model as an information transformation module to introduce auxiliary information, which can be regarded as a method based on collaborative filtering and content-based recommendation. Wang et al. (2015) focused on an important issue in the recommendation system: scoring predictions with textual information (such as blog posts, etc.). Van Den Oord et al. (2013) mainly solved the cold boot problem in the music recommendation system. In general, cold boot issues include two aspects, new users and new items, where new items are primarily considered. The recommendation algorithm of the traditional matrix decomposition performs prediction by decomposing the score into two low-rank vectors. Wang and Wang (2014) used the deep belief network to perform audio data feature transformation, except that two representations were retained at the same time. The first representation represents the data representation obtained from the collaborative filtering method, and the second part corresponds to the data representation based on the content method, and the last two parts represent the dot product separately, which is used to fit the final score result.

Based on deep learning and recommendation system application, Covington et al. proposed collaborative filtering-based deep neural network recommendation system for Youtube, and that system constitutes two neural networks, one for generating recommended videos and the other one for rankings. These two filters and its inputs decide what users can see on YouTube for recommended next video, the recommended video list, browsing video list, and so on (Covington et al. 2016). Google also proposed wide and deep learning (Cheng et al. 2016); wide and deep refers to memorization and generalization. Taking our system as an example (Fig. 2), if the user clicked on "*over memory usage*" of background app cleaning at a specific time and status on his smartphone, and our system recommended "*garbage removal*" of cleaning the app cache file, which is also accepted by the user to click, but "*the handset is too warm*" is not accepted by the user. Our system needs to record how much each pairing is preferred by the user. If we want to recommend a pop-up to users to explore similar functions, we can map it and find the closest the user might like and recommend to the user. For example, "*junk clean up*" and "*scrapbook clean up*" are very close; then, if we also recommend "*scrapbook clean up*" to users, they may also click on it because they are similar. Combining these two logics to train, the trained model will learn how to strike a balance between these two different needs.

Overall, AI technologies based on deep learning and machine learning are producing new understanding to the world. For example, based on business data, we can forecast future sales to increase sales performance and productivity. According to the forecast by IDC in 2017, the global expenditure on cognition and AI system will reach 12.5 billion US dollars with a growth rate of 59.3% comparing to 2016. The goal of this work is to describe our motivation and imple-

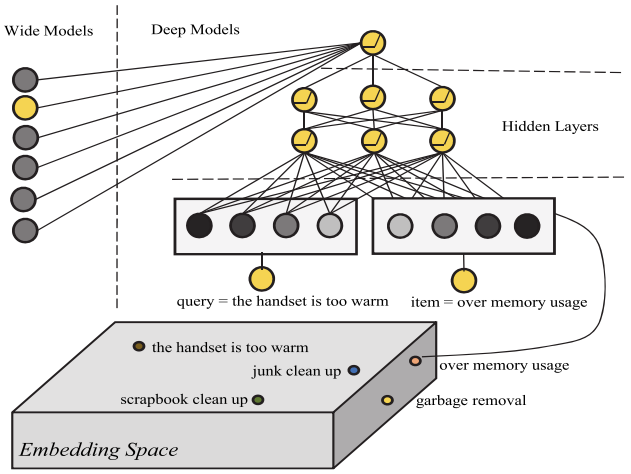

Fig. 2 Example of our wide and deep models

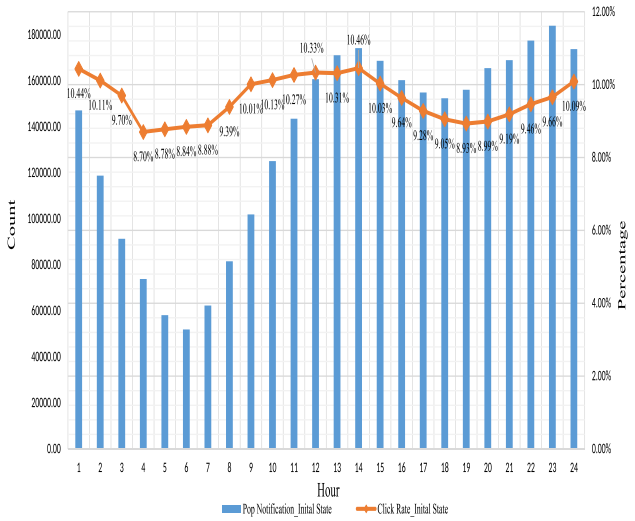

Fig. 3 Initial state of our pop notification and click rate

mentation decisions to know the challenges and strengths of the goal.

3 Our proposed system

There are three stages in developing our system: data generation, system building, and model deploying. Like all the machine learning scenarios, feature engineering is a time- and resource-consuming process. The features used will result in the performance of forecast model on click-through rate, and the potential problems and the accuracy rate of unknown data. It is not always good to have more data. Besides, due to our difference on functions from traditional category and continuity/ordering, there is a huge difference on the base number. Some are binary (for example, "is_ charging," "applock_enabled," "notification_cleaner_enabled"), and some may have millions of possible value (for example, remaining_storage, remaining_ram, storage, ram). It is more important to find out the relation of the combination of data.

The accuracy of modeling is critical metrics. Speaking of our current system, we should be more aware of the no information rate in statistics. There is about only 10% click-through rate in large sample users (as shown in Fig. 3). Using imbalance data to train the model always reflects results of users answering not recommending pop-ups. The accuracy rate reached 90%. To react to this situation, we use random forest model that can show features and feature importance to screen in advance.

There are slight differences between people in different countries or regions or even individuals. In this study, we do preprocessing based on different conditions and mechanisms of our system; meanwhile, we also classify the attributes associated with user's contextual behavior. For example, "noti_display_30" of the number of pop-ups displayed in the first 30 min, "noti_click_30" of the number of pop-ups populated in the first 30 min, "noti_display_60" , "noti_click_60" , and noti_click_last, noti_click_last2, notification_cancel_count, and other features of the number of pop-ups in the first 60 min. Through our ranking mechanism, we assign an independent score to each pop-up and feedback to the user. Our ultimate goal is to constantly adjust to real-time A/B test results. By the ranking mechanism, it can allow users to feel the pop-up notification not only based on their smartphone's usage but also to remind users at the right time to protect their smartphone with our core features.

The brief description of our hybrid model is listed as below:

- *Device layer* smartphone using condition such as remaining_storage, remaining_ram, remaining_battery, installed _day_count;
- *Process layer* Active or passive of our product functional operation such as active_scan_count, passive_scan_count, active_clean_count, passive_clean_count, active_boost_count, passive_boost_count, active_battery_saver_count, passive_battery_saver_count, applock_enabled, notification_cleaner_enabled, private_browsing_count, wifi_test_count, wifi_boost_count, notification_display_count,notification_click_count;
- *Ranking layer* Feature adjustment evaluations based on the results of pop-ups click through rate such as noti_display_30, noti_click_30, noti_display_60, noti_click_60, noti_click_120, noti_display_120, notification _cancel_count, is_null.

Figure 4 shows our system flowchart. There are six steps listed as below.

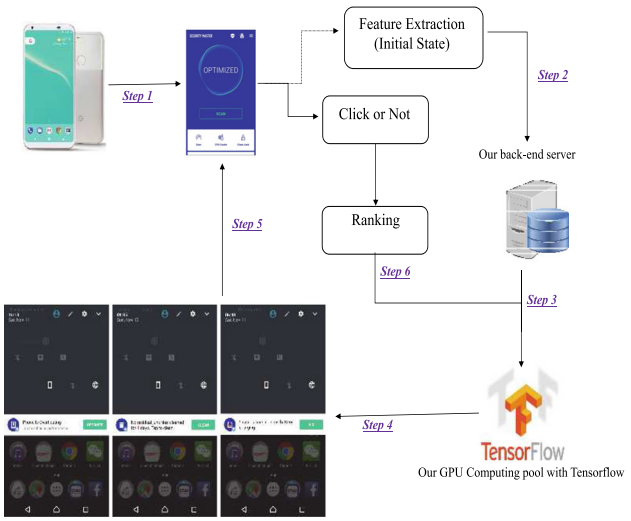

Fig. 4 Our system architecture and flowchart

- Collecting user's preference for modeling;
- Analyzing and initializing the model;
- Uploading data to backend;
- Training via Tensorflow and Keras;
- Calculating the parameters of user's pop-ups to control the number of pop-ups;
- Adjusting the model focusing on if the user clicks through the pop-ups.

A brief description of our model is shown as follows:

- The activation function is ReLU as the optimizer.
- For the first layer, the number of input neurons is 80, while the number of output neurons is 40.
- For the second layer, the number of input neurons is 40, while the number of output neurons is 20.
- For the third layer, the number of input neurons is 20, while the number of output neurons is 10.
- For the fourth layer, the number of input neurons is 10, while the number of output neurons is 5.
- For the fifth layer, the number of input neurons is 5, while the number of output neurons is 1.
- The last layer is the output layer with the sigmoid activation function.

Finally, once the model has been trained and validated, we deploy it on the backend server. For each request of the user, the server receives the retrieval options from the app terminal and scores according to the user's smartphone's uploaded status, and the time for each request is 10 ms, and then display to the user from the highest score to the lowest score.

4 System implementation and experiment

4.1 System environment

Our development environment is based on 64-bit Ubuntu 14.04, and hardware setting is 128 GB DDR4 2400 RAM and Intel(R) Xeon(R) E5-2620 v4 CPU, NVIDIA TITAN V, TITAN XP, and GTX 1080 GPUs; more specifically, the software setting is the nvidia-docker tensorflow:18.04-py2 on NVIDIA cloud. Using NVIDIA cloud docker can speed up our hardware environment and reduce the dependency of related software development. At the same time, using docker for model training does not reduce its computational efficiency, and our hardware equipment is enough for us to quickly and repeatedly test the accuracy of its predictions for optimization. Finally, considering the network environment such as the level of user amount of and countries, in order not to increase the original performance on the client side, we brought the trained model online and provide RESTful-API. The related hardware environment is g2.2xlarge on AWS EC2. We only need to upload the value of the features from client application (for example: 15,10,0,1,0,681,225,92,27,1,0,1,1,0,0,2,0,0,0,0,0,0,0,0,0,3, 1,4,13047,1024,1,2). In our experiments, the response time per inquiry is only about 10 ms, and it does not cause a delay in the reaction. This architecture can help us continue to add more notification click prediction services in the future.

4.2 Data generation

Our data are collected based on the products such as security master, clean master, and CM browser supported by Leopard Mobile Inc. (Cheetah Mobile Taiwan Agency). The Company's core products have attracted approximately 600 million global MAUs in more than 200 countries and regions, of which approximately 77% are located in Europe and the USA. This amount of data can help us repeat training and verification more accurately. We observed the growth rate of the number of pop-ups display and overall click-through rate. It is important to forecast users' preference and frequency on push notifications precisely. In this paper, we selected partial notification data and countries for our experiment. We picked up "over-used in memory" (regarded as noti 1), "mobile temperature is too high" (regarded as noti 2) and "cleaning the garbage" (regarded as noti 3) from all the notification pop-ups, to process data collection and training. Our training data are about 50,000 in total per day from all the sample countries and their respective notification columns, with clicks and no clicks in half. The tested data are collected in a scale of 1:10 based on the problem described in Fig. 3. We used a total amount of 50,000 non-repeated data in our test. According to the method described in Sect. 3, we use random forest to filter out the more important features for training and opti-

mization. Then, the model is built into the cloud and tested through real environment. The period between September 24 and 30, 2017, is regarded as Week 1, and the period between October 1 and 7 is regarded as Week 2.

4.3 Experiment result

For noti 1 as shown in Fig. 5, the result of week 1 indicated that the number of pop-ups decreased dramatically and the troublesome to users decreased simultaneously. Meanwhile, through our automatic adjustment with our ranking, the number of pop-ups continued to decrease in week 2. As shown in Fig. 6, the click-through rate in week 2 continued to increase comparing to the result from week 1. To verify whether the initial process and ranking mechanism in our system can effectively auto-adjust (as shown in Figs. 7, 8), we applied the model to noti 2 and noti 3. We found that during 2:00 AM and 7:00 AM, the number of pop-ups is similar to the number of noti 1 pop-ups. There was no significant decrease. However, the number of noti 2 pop-ups and noti 3 pop-ups in week 2 was less than the number in week 1. No matter for noti 2 or noti 3, in week 2 and week 1 between 8:00 AM and 6:00 PM as well as between 7:00 PM and 12:00 AM, the number of pop-ups is decreasing. The click-through rate increases along with the number of pop-ups.

Aside from the decrease in the number of troublesome pop-ups and increase in click-through rate, the retention rate of a smartphone mobile app is also important to maintain. Seven-day retention is a critical indicator. It represents the number of users that logs in the app at least once in the following seven days over the number of new users on that day. We found that in Fig. 9, the seven-day retention

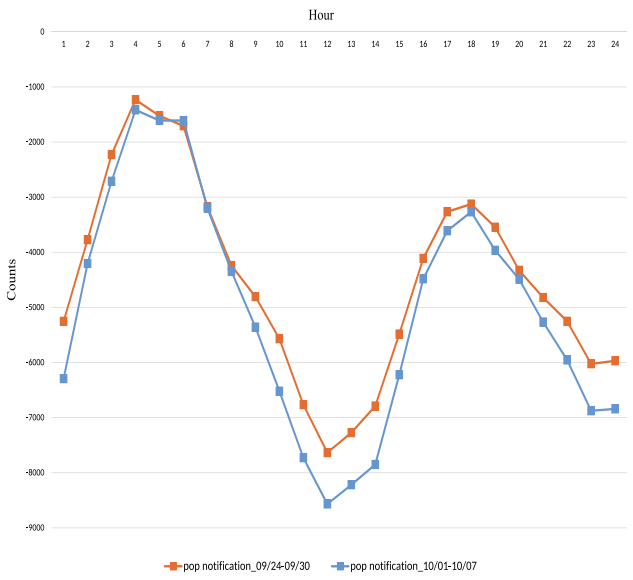

Fig. 5 Noti 1, the number of display of pop-ups

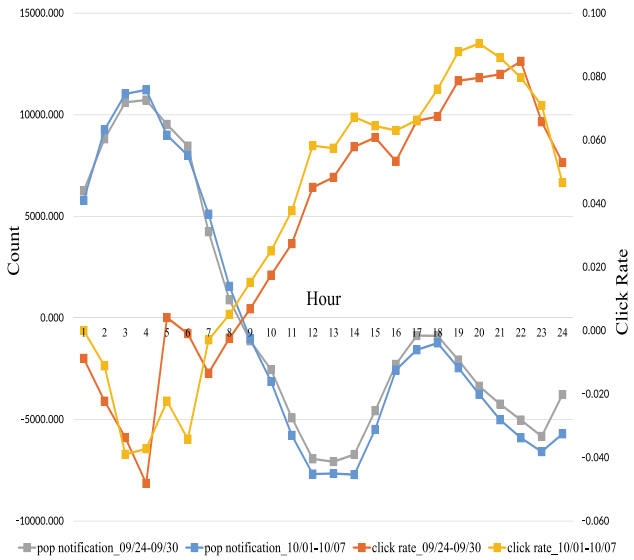

Fig. 6 Noti 1, the click-through rate of pop-ups

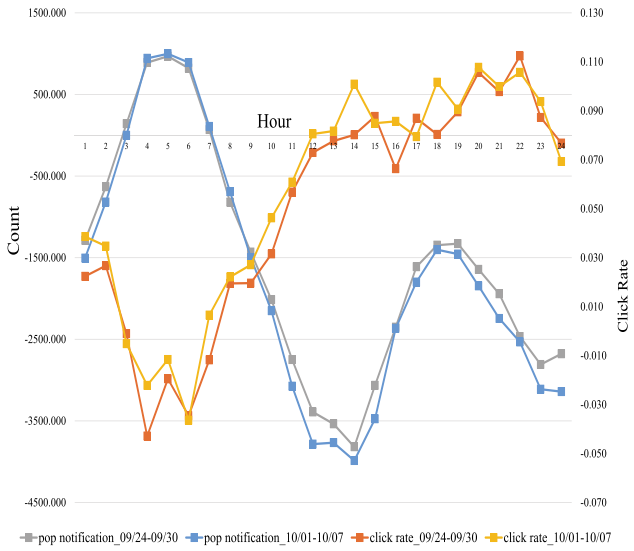

Fig. 7 Noti 2, the number of display and click-through rate of pop-ups

rate of Week1 and Week2 fluctuated between 44.05 and 44.6%. However in Week3 (2017/10/08–2017/10/14) and Week4 (2017/10/15–2017/10/21), the seven-day retention rate increased and continuously remained at about 46.7%.

5 Conclusion

We have described our proposal "***C***lick-sequence-aware dee***P*** neural network (DNN)-based ***P***op-u***P***s rec***O***mmendation (C-3PO)" and the feature engineering process. The system has been tested and verified with the products of our partners in some countries. The results showed that our system effectively decreased the number of pop-ups and increased

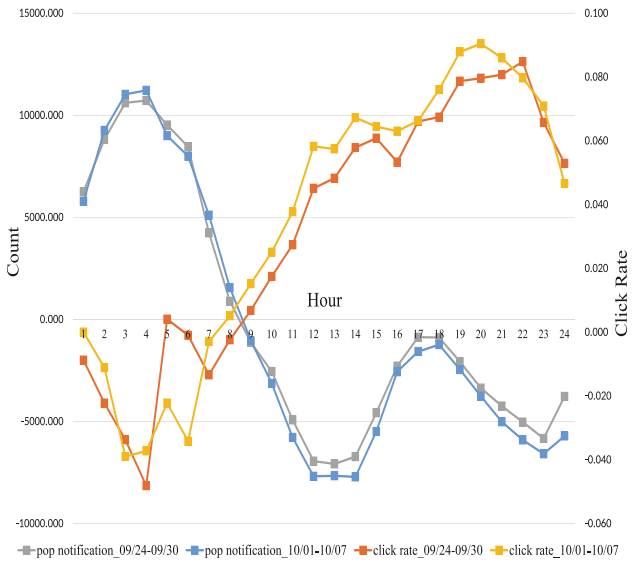

Fig. 8 Noti 3, the number of display and click-through rate of pop-ups

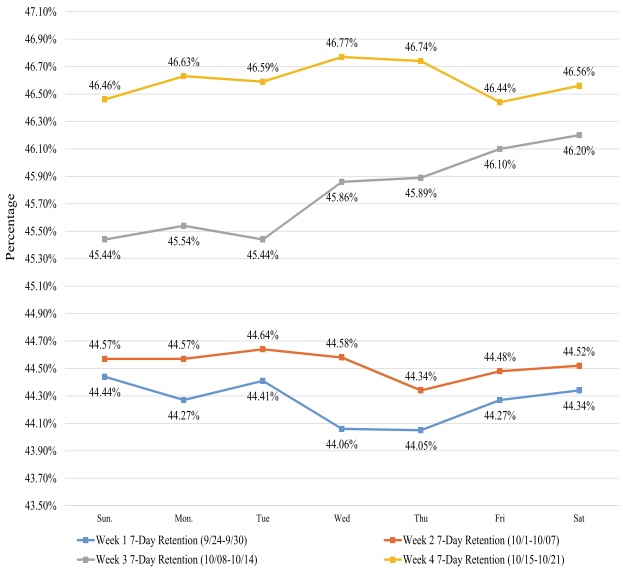

Fig. 9 Day retention of week 1–4

the click-through rate and seven-day retention rate. We now deploy the system to more products. We expect to provide more convenient user scenarios to end users or enterprises. The future work is to improve our deep learning model, to decrease complicated tasks, and to train a high-performance advertisement recommendation system. As your reference, we keep our research results and experiment material on http://C3PO.TWMAN.ORG, if there is any update.

Acknowledgements This work would not have been possible without the valuable dataset offered by Cheetah Mobile Inc.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J et al (2016) TensorFlow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation, Savannah

Aiolli F (2013) Efficient top-n recommendation for very large scale binary rated datasets. In: Proceedings of the 7th ACM conference on recommender systems. ACM, New York, pp 273–280

Cheng H-T, et al (2016) Wide and deep learning for recommender systems. In: Proceedings of the 1st workshop on deep learning for recommender systems. Boston

Covington P, et al (2016) Deep neural networks for YouTube recommendations. In: Proceedings of the 10th ACM conference on recommender systems. Boston

Goodfellow I, Bengio Y, Courville A (2016) Deep learning. An MIT Press Book, Cambridge

He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: IEEE conference on computer vision and pattern recognition (CVPR), Las Vegas

Koren Y, Bell R, Volinsky C (2009) Matrix factorization techniques for recommender systems. Computer 42(8):30–37

Krizhevsky A, Sutskever I, Hinton GE (2012) ImageNet classification with deep convolutional neural networks. In: Advances in neural information processing systems 25 NIPS. Harrahs and Harveys, Lake Tahoe, pp 1097–1105

LeCun Y, Bengio Y, Hinton G (2015) Deep learning. Nature 521:436–444

Pan R, Zhou Y, Cao B, Liu N, Lukose R, Scholz M, Yang Q (2008) One-class collaborative filtering. In ICDM, pp 502–511

Salakhutdinov R, Mnih A, Hinton GE (2007) Restricted Boltzmann machines for collaborative filtering. ICML, pp 791–798

Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G et al (2016) Mastering the game of Go with deep neural networks and tree search. Nature 529:484–489

Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In: International Conference on Learning Representations 2015 (ICLR2015), San Diego

Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR), WA

Van Den Oord A, Dieleman S, Schrauwen B (2013) Deep content-based music recommendation. NIPS, pp 2643–2651

Verstrepen K, Goethals B (2014) Unifying nearest neighbors collaborative filtering. In: Proceedings of the 8th ACM conference on recommender systems. ACM, New York, pp 177–184

Wang X, Wang Y (2014) Improving content-based and hybrid music recommendation using deep learning. ACM Multimedia, pp 627–636

Wang H, Wang N, Yeung D-Y (2015) Collaborative deep learning for recommender systems. KDD, pp 1235–1244

Wu Y, DuBois C, Zheng AX, Ester M (2016) Denoising auto-encoders for top-N recommender systems. WSDM, pp 153–162

Zheng Y, Tang B, Ding W, Zhou H (2016) A neural autoregressive approach to collaborative filtering. arXiv:1605.09477