International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Uniformity of Batch Job Recommendations and Real-Time Recommendation

Sajid M. Momin[1], Sahil Ajamuddin Mujawar[2], Sushil Bajirao Magadum[3], Abhijeet Paritekar[4]

[1]Assistant Professor, Bharti Vidyapeeth college of Engineering, Kolhapur
[2,3]HOD, Dr Bapuji Salunkhe Institute of Engineering and Technology
[4]System Analyst, Dr Bapuji Salunkhe Institute of Engineering and Technology

Abstract

Collaborative filtering recommends items based on similarity measures between users and/or items. The basic assumption behind the algorithm is that users with similar interests have common preferences. There are a lot of applications where websites collect data from their users and use that data to predict the likes and dislikes of their users. This allows them to recommend the content that they like. Recommender systems are a way of suggesting similar items and ideas to a user's specific way of thinking. Content based recommendation system is supervised machine learning used to induce a classifier to discriminate between interesting and non-interesting items for the user. Collaborative filtering recommendation systems are basically trained in a batch manner and are designed to produce personalized recommendations for a large number of users at the same time. However, in many industrial use cases, it is reasonable to produce recommendations in real time, taking account of very recent user interactions. In this work, we present the implementation of batch and real-time recommendation system. We believe that our results can help other organizations to take informed decisions about whether to make the effort of moving from a batch to a real-time recommendation setting.

Keywords: collaborative filtering, job recommendations, real-time recommendations

I. Introduction: -

In Collaborative filtering, we are able to find similar users and recommend what similar users like. In this type of recommendation system, we not use the features of the item to recommend it, rather we classify the users into clusters of similar types and recommend each user according to the preference of its cluster. There are basic four algorithms types to build Collaborative filtering based recommender systems:

- Memory-Based
- Model-Based
- Hybrid
- Deep Learning

Persons	Movie A	Movie B	Movie C	Movie D
Person P1	5	3		5
Person P2	4		3	
Person P3	1	1		4
Persons P4	0	2		1

In this above of example, we can see that Person 1 and Person 2 give nearly similar ratings to the movie, so we can conclude that Movie C is also going to be averagely liked by Person 1 but Movie D will be a good recommendation to Person 2, like this we can also see that there are users who have different choices like Person 1 and Person 3 are opposite to each other. One can see that Person 3 and Person 4 have a common interest in the movie, on that basis we can say that Movie D is also going to be disliked by Person 4. This is Collaborative Filtering, we recommend to users the items which are liked by users of similar interest domains.

II. RELATED WORK

Recommendation methods for millions of customers, items are successfully use in the industry for at least 27 years. In 1998, Amazon launched item-based collaborative filtering [3], [8]. Recommendation systems gained attention in 2007 when Netflix's organized the Netflix Price competition and offered one million dollars for creating a recommendation model outperforming the Netflix algorithm by 10% in terms of RMSE [2]. In 2016, Netflix described their recommendation systems and announced that 80% of hours streamed at Netflix come from recommendations [11]. Recommendation systems usually apply processes consisting of two phases: training the model and generating the recommendations [6]. We can distinguish three types of approach with respect to their behaviour when the user performs an action like following cases

Case 1- the model IS NOT retrained, but the user recommendations are Influence
Case 2- the model IS NOT retrained and the user recommendations are not Influence
Case 3- the model IS retrained and the user recommendations are Influence

The first case, which we refer to as batch recommendations is the cheapest and the easiest to develop. This solution is good when a large number of recommendations are generated at the same time (e.g., for sending email recommendations to all users).

In the second case, which we refer to as real-time recommendations the most recent user interactions impact the recommendations for the user in question, but do not impact the recommendations produced for other users (because the model is not change). For that moment, the model may consist of item embedding learned once a day, whereas the user representation can be calculated on demand as an average of the representations of the items with which the user interacted.

In the third case, the model constantly learns from a continuous stream of interactions. Such approaches, known as stream-based recommender systems require the greatest engineering effort. Even in the case of collaborative filtering, these methods are able to recommend an item after some user has interacted with it.

III. BATCH AND REAL-TIME RECOMMENDATIONS

Solution	Developer	Type	Description
Storm	Twitter	Streaming	Twitter's new streaming big-data analytics solution
S4	Yahoo!	Streaming	Distributed stream computing platform from Yahoo!
Hadoop	Apache	Batch	First open source implementation of the MapReduce paradigm
Spark	UC Berkeley AMPLab	Batch	Recent analytics platform that supports in-memory data sets and resiliency
Disco	Nokia	Batch	Nokia's distributed MapReduce framework
HPCC	LexisNexis	Batch	HPC cluster for big data

A significant application of real-time processing is within embedded systems. These are specialized computing systems embedded within larger devices, and they play crucial roles in various industries, from automotive to medical and beyond. Within these contexts, the demands are often high for swift sensor data acquisition and prompt data processing. For example, a vehicle's braking system might rely on real-time processing to detect sudden obstacles and command an immediate brake, or a medical monitor might need to provide live feedback on a patient’s vitals, triggering alarms if any irregularities are detected.

Batch data processing involves gathering substantial volumes of data over a period and then processing the accumulated data (batch) in one go. Rather than handling each data record individually as they are received, the system holds and batches them together for a combined processing run

Challenges in BATCH AND REAL-TIME RECOMMENDATIONS:-

- High system complexity: In applications that demand massive scalability, businesses often invest heavily in sophisticated and costly infrastructure systems to cater to instantaneous user responses. Complex Scheduling: Real-time systems often have to juggle multiple tasks with different priority levels and timing constraints. Crafting a scheduling strategy that can handle these tasks and meet all deadlines can be complex. Some of these scheduling strategies include Round Robin and First in First out (FIFO).
- Flexibility of handling data variations: Batch processing systems typically anticipate that all incoming data is uniform, expecting them to undergo identical processing steps. If there are variations in the data, it can potentially lead to computational errors.
- Delayed Processing: Contrary to real-time and stream processing, batch processing accumulates a substantial amount of data before initiating the processing. This approach can be inefficient when an instantaneous response is crucial for the user

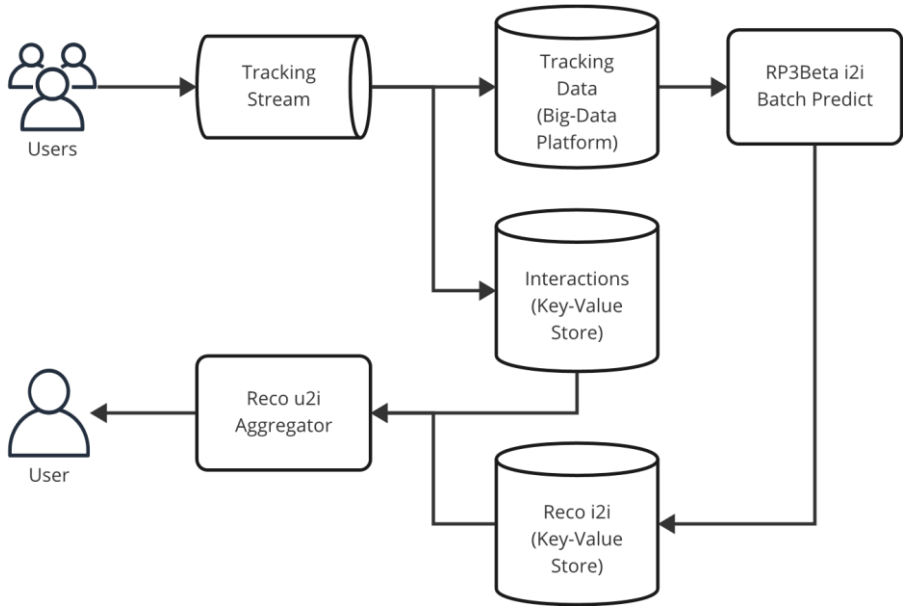

Figure 1: Real-time recommendations architecture

- Most recommendation models, not necessarily graph-based, are able to provide item-to-item recommendations in a batch mode (for all the items at the same time). Hence we can utilize the architecture is to provide realtime recommendations based on the user's latest interactions with these items. For some models, such recommendations may not be the same as the user-to-item recommendations produced by the model in a usual way (defined within the model). For instance, in matrix factorization methods [31], the recommendations are calculated based on user embedding learned during the training procedure. Two users with the same set of interactions might have different embedding, hence different recommendations produced in the usual way. In our approach, such users would receive exactly the same recommendations.
- Recommendations produced by our approach are identical (for large enough values of M and N) to recommendations produced in a usual way for models where user-to-item recommendations can be obtained by aggregating item to-item recommendations.

IV. Observations

- In this work, we described the batch and real-time recommendation. We discussed the architectural aspects of these approaches. Then we provided the results of online A/B tests conducted on OLX users. We reported that batch model with the real-time search model increases the number of users replying to recommended job ads by at least 10 %. Even though our observation were conducted using the some example model, we can use any recommendation system if the recommendations for a user can be calculated based on items similar to the items with which that user has interacted. Additionally, we believe that in domains where users are more likely to change their preferences, the impact of utilizing a real-time recommendation system may be even greater.

V. References

1. A. Sharma, J. Jiang, P. Bommannavar, B. Larson, and J. Lin, ''GraphJet: Real-time content recommendations at Twitter,'' *Proc. VLDB Endowment*, vol. 9, no. 13, pp. 1281–1292, Sep. 2016.
2. J. Bennett and S. Lanning, ''The Netflix Prize,'' in *Proc. KDD Cup Workshop*, New York, NY, USA, 2007, p. 35.

3. B. Smith and G. Linden, ‘‘Two decades of recommender systems at Amazon.com,’’ *IEEE Internet Comput.*, vol. 21, no. 3, pp. 12 –18, May 2017.
4. R. Kwieciński, T. Górecki, and A. Filipowska, ‘‘Learning edge importance in bipartite graph-based recommendations,’’ in *Proc. Ann. Comput. Sci. Inf. Syst.*, Sep. 2022, pp. 227–233.
5. R. Chen, Q. Hua, Y.-S. Chang, B. Wei, L. Zhang, and X. Kong, ‘‘A survey of collaborative filtering-based recommender systems: From traditional methodstohybridmethodsbasedonsocialnetworks,’’*IEEEAccess*,vol.6, pp. 64301–64320, 2018.
6. B. Chandramouli, J. J. Levandoski, A. Eldawy, and M. F. Mokbel, ‘‘StreamRec: A real-time recommender system,’’ in *Proc. ACM SIGMOD Int. Conf. Manage. data*, New York, NY, USA, Jun. 2011, pp. 1243–1246.
7. A. D. Viniski, J. P. Barddal, A. D. S. Britto Jr., F. Enembreck, and H. V. A. D. Campos, ‘‘A case study of batch and incremental recommender systems in supermarket data under concept drifts and cold start,’’ *Expert Syst. Appl.*, vol. 176, Aug. 2021, Art. no. 114890.
8. G. Linden, B. Smith, and J. York, ‘‘Amazon.com recommendations: Itemto-item collaborative filtering,’’ *IEEE Internet Comput.*, vol. 7, no. 1 , pp. 76–80, Jan./Feb. 2003.
9. M. Al-Ghossein, T. Abdessalem, and A. Barré, ‘‘A survey on stream-based recommender systems,’’ *ACM Comput. Surv.*, vol. 54, no. 5, pp. 1 –36, May 2021.
10. M. Jugovac, D. Jannach, and M. Karimi, ‘‘StreamingRec: A framework for benchmarking stream-based news recommenders,’’ in *Proc. 12th ACM Conf. Recommender Syst.*, New York, NY, USA, Sep. 2018, pp. 269–273.