A Unified Recommendation Framework for Zoho CRM

Author information scrubbed for double-blind reviewing

No Institute Given

Abstract. A unified recommendation service is a one-stop solution for recommendations across different downstream tasks. In a dynamic platform like Zoho CRM where each customer has their dataset configurations for which different kinds of recommendations are provided, a unified system of recommendation engine is essential. However, unified systems suffer from many problems, including a lack of easy customization, explainability, etc. Moreover, each downstream task requires its catalogue of models and the best model has to dynamically change based on the dataset.

In this paper, we developed a framework to solve recommendations across multiple downstream tasks. The proposed framework decides the best model across a catalogue of models ranging from traditional methods to attention-based models for each downstream task using the ranking function determined by the combination of our custom data splitter and evaluation metrics. For explainability, we discuss a unique way to identify matching attributes of users with similar preferences. We also present our recommendation reasoning model that provides insights about users, products, and recommendations that are augmented by generative AI. This framework has been incorporated as a recommendation builder in Zoho CRM.

Keywords: Recommendation · Deep Learning · Scalability · Explainability

1 Introduction

A recommender system is used to identify and analyze user data such as their interests, purchase details, behavioral patterns, etc. to suggest the most relevant product to them. It also compares the behavioral patterns and historical data of other similar customers to aid recommendations. With the help of such recommendation systems, businesses can bring about significant impacts on their revenue, sales, conversions, click-through rates, etc. Businesses can also use these recommendations and tailor their marketing strategy to create personalized ad campaigns. Recommendation systems are used to solve wide-ranging problems in platforms such as e-commerce websites, customer relationship management [7] (CRM), etc. While the core aspects of these systems remain the same, there are subtle differences in the expectations and use cases when used in a CRM platform

as compared to an e-commerce platform [14]. A salesperson using CRM expects concrete reasoning behind a recommendation to approach, justify, and convince their lead about the product. Therefore the recommender system for a CRM is generally held to higher and stricter standards, with the necessity for explainable recommendations and insights on the user, product, and the recommendation provided.

The Zoho CRM [1] platform has helped transform businesses across all sizes. Supporting businesses throughout the size spectrum has its challenges. While some businesses are well established with thousands of employees and millions of interactions, the same cannot be expected throughout our customer base. The size of the transaction dataset required to train our recommendation models varies drastically. A cutting-edge state-of-the-art deep learning model could work well for large customers with millions of transactions but might find it challenging to provide accurate recommendations for small customers with interactions in the thousands. For those customers where enough data is not present to train deep-learning based and other lossy models, lossless algorithms like pattern mining could work well. A one-size-fits-all approach can never work here. Moreover, we cannot analyze the nature of the interactions and pick a model manually for each customer. There is a need for dynamic model selection based on the transaction dataset. Traditional recommender systems generally do not provide reasoning, which also poses a problem and warrants the requirement for a custom solution.

As there are no recommendation approaches that universally fit all transactional data across diverse customer bases, we developed a framework (fig. 1) to train a catalogue of algorithms for a common train split. We then evaluate the algorithms on a carefully designed test split to ensure the comparisons are as fair as possible. The best model is identified and hosted on our inference endpoints. The model selection process is dynamic and a training job runs in the background to reflect on new data changes and determine if the incumbent model still provides the best results. If the incumbent model is outperformed by a candidate model, the candidate model is now hosted on the inference endpoints. The training process for the framework can be configured to run periodically or run every time a significant change in data is detected. For insignificant changes in data, the incumbent model alone is retrained on the new data to provide updated recommendations. This ensures that our customers get the best recommendations possible irrespective of the underlying model.

In a sales pitch, a salesperson often establishes social proof by referencing other customers who bought a particular product and explaining the similarities between them to make it easier for prospective buyers to trust their recommendations. To assist our CRM customers in their sales process, we developed an approach to identify reference customers who are relevant to the recommendation made and highlight similar and matching attributes between them and their target lead. This approach is independent of the algorithm used to arrive at the recommendations. In addition to reference customers, we also provide generative AI powered insights on the user, product, and the recommendations made.

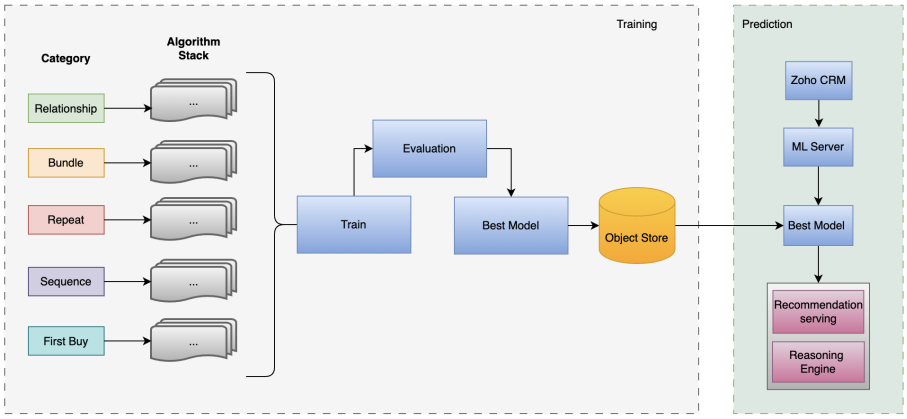

Fig. 1. Unified Recommendation Framework Architecture

All these features help us provide concrete reasoning and insights to our sales customers, leading to better conversion rates and other metrics.

The rest of the paper is organized as follows. In the next section, we discuss the framework by explaining a few of the downstream tasks and models that we support at Zoho CRM, followed by our custom data splitters, evaluation metrics, and ranking. We end the section by discussing our novel approach to explaining recommendations. In section 3, we move on to results and discussions and finally conclude the paper in section 4.

2 FRAMEWORK DISCUSSION

We now discuss in detail about a few of the downstream tasks and models that are supported in Zoho CRM as well as the data splitters, evaluation metrics and ranking algorithm.

2.1 DOWNSTREAM TASKS

A recommendation system can be configured to perform recommendations on many downstream tasks. In this paper, we evaluate our framework for the following downstream tasks.

First Purchase The first purchase downstream task is designed to provide recommendations on products or services to users who are yet to make their first purchase on the platform. Typically, users who fall under this category have limited or no prior data. The first purchase algorithms rely heavily on user features such as their demographics, interests, and other profile information. The

goal of this downstream task is to improve user engagement on the platform by encouraging new users to make their first purchase and establish long-term relationships.

Next Purchase The next purchase downstream task suggests products based on a user's prior purchases and interactions on the platform. The next purchase algorithms take into consideration the user features as well as their prior purchase sequence to predict the product that they are likely to buy next. By providing next purchase recommendations, this downstream task aims to drive a continuous purchase cycle and increase user engagement on the platform.

Repeat Purchase The repeat purchase downstream task aims to provide recommendations that encourage repeat purchases by understanding the interaction data of users who often purchase a certain product or service. The repeat purchase algorithms first identify products that are repeatedly bought by the user. It then analyses patterns such as frequency and timestamps from prior purchases to predict when the customer is likely to buy the product again. By predicting the time of the next purchase, it provides timely recommendations to the user to keep them engaged and drive repeat purchase cycles on the platform.

Relationship and Bundle The relationship downstream task suggests additional products to users based on their prior purchase or current selection. The algorithms analyze the purchasing patterns of the users and identify items that are bought together. When a user selects a product for purchase, the algorithms generate recommendations for other products that previous customers bought that are relevant to the selected item. The bundle downstream task differs from relationship in how the interaction data is analyzed. Identification of items that are bought together has an additional constraint of a 24-hour time interval. The remaining aspects remain unchanged. Both of the downstream tasks are designed to help increase sales by encouraging users to purchase complementary items and enhance platform engagement.

Popular and Trending Popularity is a measure of the number of interactions at a certain time interval, whereas trending in the recommendation context can be considered as the rate of change in popularity, or acceleration. The primary aim of trending downstream algorithms is to identify and promote items that are increasingly purchased currently, while not necessarily being products that are popular on a global time scale. By promoting the right items at the right time, the downstream task aims to capture purchase sentiments and increase sales and engagement.

Personalized for You Personalized for you downstream task provides tailored suggestions or recommendations to individuals based on their unique preferences, behaviors, demographics, and past interactions. By leveraging advanced

algorithms, personalized recommendation systems understand user preferences and predict items they are likely to be interested in. It enhances user experience, increases engagement, and drives conversion rates for businesses by delivering relevant and timely suggestions to individual users.

2.2 Framework supported algorithms

Zoho CRM uses a variety of algorithms from traditional to graph-based models. SOTA deep learning approaches tend to perform well on large interaction datasets whereas traditional machine learning and factorization-based approaches work well on small datasets. This flexibility enables us to dynamically select the best model irrespective of the data size. A few of the popular algorithms and models used in Zoho CRM along with the downstream tasks are listed in the table. 1:

User Co Purchase Matrix The User Co Purchase Matrix approach utilizes the user-item interaction data to construct an item co-occurrence matrix [10]. The matrix element at (i, j) represents the number of instances where items i and j occurred together. The sparse matrix can be constructed by grouping the interactions based on users and analyzing their purchases. If the user purchases a set of 'n' products, these products are said to co-occur and the corresponding elements in the sparse matrix are incremented.

Transaction Co Purchase Matrix The Transaction Co Purchase Matrix approach is almost identical to the user co purchase matrix, with the difference lying in how transactions are grouped. Transactions are grouped based on purchase timestamp instead of users, making it suitable for bundle recommendations as items purchased as bundles have the same purchase timestamp.

Biased Random Walk Biased random walk [18] approaches are gaining popularity in recommendation systems. Systems such as Pixie [5] have demonstrated the scalability and speed benefits associated with graph data structures. Users and items are mapped as nodes in the graph, with the edges representing the transactions. Given an input item and a user, random walks are performed from the input item node. A bias is applied to the connecting edges based on how similar a user node is to the input user, pushing the walk toward item nodes that are purchased by users with similar preferences. This makes it suitable for bundle and relationship recommendation tasks.

Deep Interest Network The Deep Interest Network [19] is an efficient approach to processing user behavior data. It uses Local Activation Units that are heavily inspired by the attention model [2] to dynamically generate user representation vectors. These vectors are calculated by taking into consideration past purchases to model user interests. Shifting away from fixed length to

adaptive length representations for capturing user interests eliminates the bottleneck problem found in traditional CTR models. DIN is naturally suited to sequence predictions where the next purchase is predicted based on the user's prior purchases.

SuBSeq Succinct BWT-Based Sequence Prediction [8] is designed specifically for sequential recommendation. It utilizes the Burrows-Wheeler Transform and leverages advanced techniques to extract meaningful patterns from interaction data. It focuses on sub-sequence mining to identify recurring sequences and intricate patterns within user behavior sequences. It excels in capturing temporal dependencies, user preferences, and contextual nuances through a combination of efficient data processing and pattern recognition to provide highly personalized recommendations tailored to individual behaviors while remaining lightweight in terms of memory and computational complexity.

LightFM LightFM [9] is a robust recommendation algorithm that leverages user profiles, item details, and interaction data to provide personalized recommendations. By merging collaborative filtering with matrix factorization techniques, LightFM adeptly captures user preferences and item features, and by analyzing user profiles and item characteristics, it delivers relevant recommendations. It simplifies the discovery of related products and adjusts recommendations promptly based on user behavior. With insights into user demographics, preferences, and interaction history, LightFM enhances recommendation experiences across diverse industries and applications, fostering engagement and satisfaction.

DeepFM Deep Factorization Machine [6] is a state-of-the-art recommendation algorithm that combines the strengths of factorization machines [13] with deep neural networks [15]. It uses factorization techniques to learn low-dimensional embeddings for users and items while simultaneously leveraging deep neural nets to model higher-order features. This allows it to integrate both linear and non-linear features to capture complex patterns in the data and provide accurate recommendations. Its hierarchical architecture allows it to efficiently handle large-scale datasets and achieve superior performance in recommendation tasks compared to traditional methods.

Wide and Deep The Wide and Deep [4] recommendation algorithm combines linear models with deep neural networks to address the challenges of memorization and generalization. The wide component leverages linear models to memorize sparse features, enabling it to capture relationships between frequently occurring features. The deep component utilizes neural networks to learn feature embeddings and high-level representation. The neural network enables it to capture complex and non-linear patterns between the features. The deep component also enables the model to generalize on unseen real-world data.

Recurrence Finder Recurrence Finder identifies and predicts recurring events, including customer product purchases, event attendance, etc. by leveraging historical timestamps to forecast future occurrences. It assists in optimizing strategies for customer retention, and event and time management tools. This makes it a perfect candidate to provide recommendations for the Repeat Purchase downstream task [3].

Hierarchical Navigable Small Worlds Hierarchical Navigable Small Worlds [11], or HNSW is a popular algorithm based on graph data structures and skip lists [12]. It is designed to perform efficient nearest neighbors search on large datasets in higher dimensional spaces. Its hierarchical architecture facilitates rapid and scalable searches by reducing the need to traverse the entire dataset, leading to quicker query responses.

Knowledge Graph Attention Network Knowledge Graph Attention Network [16] (KGAT) is a structured graphical representation of users and items. In addition to capturing user-item interactions as edges, it also captures how items relate to each other through the recursive embedding propagation process. It iteratively updates a node's embedding by considering the embeddings of its neighbors in the knowledge graph. This effectively captures the complex relationships between items. Furthermore, KGAT employs an attention mechanism to assign weights to these neighboring nodes. This ensures that KGAT focuses on the most relevant connections within the knowledge graph, leading to more accurate and informative item representations.

Light GCN Light GCN is a simplified type of graph convolutional network designed to provide recommendations based on collaborative filtering approach. It removes many of the complexities associated with traditional graph convolutional networks and mainly focuses on propagating user-item interactions through the graph. It computes node embeddings by iteratively aggregating neighboring embeddings. The simplified nature of Light GCN allows it to be highly scalable and suitable for large-scale datasets while simultaneously providing competitive recommendations compared to other graph convolutional networks.

2.3 DATA SPLITER

Each category or downstream task requires different ways to split the transactional dataset into training and evaluation sets. The data flow is handled using big data tools. We use Presto as our engine to query the required datasets from our CRM databases. The data is then persisted on the Hadoop File System (HDFS). This acts as the data source for the framework. The data splitter approach for each category is discussed below:

Table 1. Downstream tasks and models

Downstream Task	Model
First Buy	Light FM Deep FM Wide and Deep HNSW
Next Buy	DIN SuBSeq
Relationship	User Co Purchase Matrix Biased Random Walk KGAT
Bundle	Transaction Co Purchase Matrix Biased Random Walk KGAT
Repeat Purchase	Recurrence Finder
Personalized for you	Light FM Light GCN KGAT

First purchase This downstream task is exclusively created to generate recommendations for cold-start users. For training the recommender model, we only consider the first transactions of all the users along with their properties and the product properties. 70% of the transaction dataset is used for training and the remaining 30% is used for evaluation. Only the transaction dataset is split to maintain input consistency, all user and product properties are sent during training since certain algorithms require user features to make first-buy predictions

Repeat purchase If a customer has purchased a product n times, n - 1 purchases are considered for training. The dataset contains the products as well as the time of purchase. The algorithms are evaluated with the nth purchase. Then the model is retrained with all n purchases and then made to predict the date of its next purchase. If there are no users with repeat purchases found, the downstream activity is skipped.

Relationship Firstly, all transactions are grouped by user. The grouped transaction dataset is then divided into training and evaluation sets, with 70% going towards training and the remaining towards evaluation. Users with only one transaction are automatically included in the training set. In the evaluation set, a random product in the grouped products for the respective user is kept as the prediction input, and the rest of the products bought are considered as ground truth for evaluation.

Bundle Unlike the previous downstream task where all transactions are grouped by users, there is a 24-hour purchase window for grouping. Users with only one transaction are included in the training set and the remaining grouped transaction set is split into training and evaluation sets with the same ratio as above. Similar to the previous downstream task, a random product from the grouped products for the respective user is selected as prediction input with the remaining products considered as ground truth for evaluation.

Sequence The transaction data is organized chronologically based on transaction timestamps and then grouped by individual users. The grouped transaction dataset is split into training and evaluation sets with the same ratio as above. For evaluation, the most recent purchase is concealed and used as ground truth, and the predictions are made on the remaining purchases.

2.4 EVALUATION METRICS

In our framework, we use widely accepted evaluation metrics such as accuracy, precision, recall, normalized discounted cumulative gain, coverage, etc. to evaluate the performance of all the algorithms across all downstream tasks.

First Purchase and Sequence Both the first purchase and sequence downstream tasks focus on predicting a single product or service. While the first purchase downstream task predicts the initial purchase of a new user, the sequence downstream task predicts the immediate next purchase for a given purchase sequence. Both algorithms share the following same evaluation metrics:

- **Accuracy**: A metric to assess the percentage of correct predictions.
- **Normalized discounted cumulative gain [17]**: A metric to assess a recommender system's performance by considering the products recommended as well as their position in the recommendation ranking.
- **Coverage**: A metric to assess how diverse the recommendations are.

Relationship, Bundle and Personalized Recommendation For these algorithms, the recommendations comprise multiple products, which are evaluated against ground truth products. All three algorithms use the following evaluation metrics:

- **Precision**: A metric to measure the amount of relevant recommendations.
- **Recall**: A metric to measure the recommendation model's ability to capture all relevant products in its recommendations.
- **Normalized discounted cumulative gain**
- **Coverage**

Repeat Purchase For repeat purchases, the recommendation model predicts when a user purchases a specific product again by factoring in their historical purchase intervals. The evaluation metrics for this category of algorithms are as follows:

- **Recurrence Rate**: A metric to measure how often a given product is purchased by a user.
- **Mean Absolute Error**: A metric to measure the absolute difference between actual and predicted purchase intervals.
- **Mean Squared Error**: A metric to measure the squared difference between actual and predicted purchase intervals.

2.5 SCALABILITY METRICS

While evaluation metrics perform a reasonable job of identifying the performance of given models, it is unwise to only consider them. A very important consideration to be made while deciding on a particular algorithm for production is the time taken to run inference. An accurate model that takes a long time to provide predictions might not be an appropriate choice for time-critical applications. In the case of a CRM solution, customers must be served with instant recommendations. We use the following metrics to determine the scalability of all the algorithms in the catalogue:

- **Training time**: Total time taken to train a particular algorithm.
- **Prediction time**: Total time taken to run inference on a particular algorithm.
- **Peak memory consumption**: Peak memory usage during training

2.6 RANKING

Upon completion of training for all the algorithms in the catalogue within each downstream task, we possess a comprehensive set of evaluation and scalability metrics. We then assign weights to each metric type and calculate a score. The weights can either be learned using a separate algorithm or can be assigned based on the requirements. Weights for scalability metrics can be low or high depending on the time-sensitive nature as well as the hardware resources available for the particular use case. This score serves as a basis for selecting the optimal model for a given category. We utilize object stores to store the optimal model and use it for subsequent predictions. The ranking algorithm lets us identify the most suitable algorithm for a particular dataset and use case.

2.7 EXPLAINABILITY

Reference Customer CRM users, especially salespeople rely on concrete reasoning behind the recommendations to prepare their pitch to prospective leads. A recommendation system must not only provide product recommendations but

must also be able to provide the rationale behind its selection. To achieve this, we identify matching customers with similar attributes to a current user to justify product recommendations. The identification process is done using two indices: **product inverted index** and **user inverted index**:

- The product inverted index contains a mapping between each product and a list of users who bought that particular product
- The user-inverted index contains a mapping between each feature value and a list of users who share the same feature values.

Given a product recommendation and a user, the algorithm first retrieves users who purchased the recommended product from the product inverted index. Using user inverted index, the algorithm identifies and assigns matching scores for users who purchased the same product based on the feature lists. In other words, for each feature of a given user, compare the feature values of users who purchased the same product and assign scores based on the match. The top users are considered reference customers and can be returned to the user along with their attributes to justify recommendations.

Custom Insights In addition to identifying reference customers, insights are derived on the users, products, and recommendations provided to further explain the recommendations using the following metrics.

- **User Transaction Score**
  The user transaction score is derived from the transaction data to assess an individual's spending behavior using two metrics:
  - The transaction contribution of user i, TC_i, represents the fraction of total transactions contributed by user i

$$TC_i = \frac{\text{transaction count of user } [i]}{\text{total transaction count}} \tag{1}$$

$$TC_{avg} = \frac{1}{n} \sum_{i=0}^{N} \frac{\text{transaction count of user } [i]}{\text{total transaction count}} \tag{2}$$

  - The importance score of user i, IS_i, represents the relative transaction contribution of i with respect to other users.

$$Z_i = \frac{TC_i}{TC_{avg}} \tag{3}$$

$$IS_i = \frac{e^{Z_i}}{\sum_{i=0}^{N} e^{Z_i}} \tag{4}$$

  The average of transaction contribution and importance scores represent the final user transaction score and it provides a quantitative measure of the user's engagement in overall transactions.

$$\text{transaction score for user i} = \frac{TC_i + IS_i}{2} \tag{5}$$

– **Product Transaction Score**
We use the same metrics and equations to quantitatively measure the product's engagement in overall transactions.

$$\text{transaction score for product i} = \frac{TC_i + IS_i}{2} \tag{6}$$

– **Product Choice Score**
Product i's choice score PS_i represents the number of unique users who purchased the product relative to the total user base.

$$PS_i = \frac{\text{count of unique users who bought product A}}{\text{total user count}} \tag{7}$$

The product choice score is an indicator of how attractive and reachable the product is to users.

– **Trending Score**
The trending score is a dynamic measure evolving over time to capture a product's momentum in the market. To calculate a product's trend score, we measure the z-score of the number of products sold in the normal distribution plot of the exponential moving average and perform softmax to determine relative growth/decline in sales.

– **Impact Score for Ignored Recommendations** The impact score for ignored recommendations aims to quantify the potentially missed opportunity when a recommendation is ignored. The impact can be calculated by analyzing the volume of transactions that would have involved the recommended product as well as the users who would have engaged with the recommended product. It can be modeled as the average of transaction score and product choice score for product i.

$$\text{product i impact score} = \frac{TC_i + PS_i}{2} \tag{8}$$

3 RESULTS AND DISCUSSIONS

To test the framework, we ran training and inference on more than 100 customer datasets. Due to varying interaction data sizes, some models perform better for a particular dataset than others. Table 2 provides a brief overview of the dataset from one of our customer configurations. The number of users and transactions suggests that Factorization models perform better as deep learning models tend to struggle with less data. Table 3 shows the performance of models on first purchase and sequence tasks and unsurprisingly, LightFM performed the best whereas deep learning based DIN, DeepFM, and Wide Deep models performed poorly. In table 4, we present the results for relationship and bundle downstream tasks and it is close between co-purchase matrix-based methods and graph-based models for this particular dataset. The best model in this case will be decided by the scalability metrics. The weights associated with individual scalability metrics determine how important each metric is in the ranking function, and these

weights are governed by the use case. Fig. 2 shows the distribution of training time across configurations for different downstream tasks. For background jobs, accuracy could be more important than training and inference times. For hardware-limited customers, peak memory consumption and model size could be crucial. Ultimately, these weights are customizable by the end-user depending on their requirements. The framework identifies the best model for a particular customer based on the ranking function. By not fixating on a particular model, we ensure the best possible recommendations for our customers.

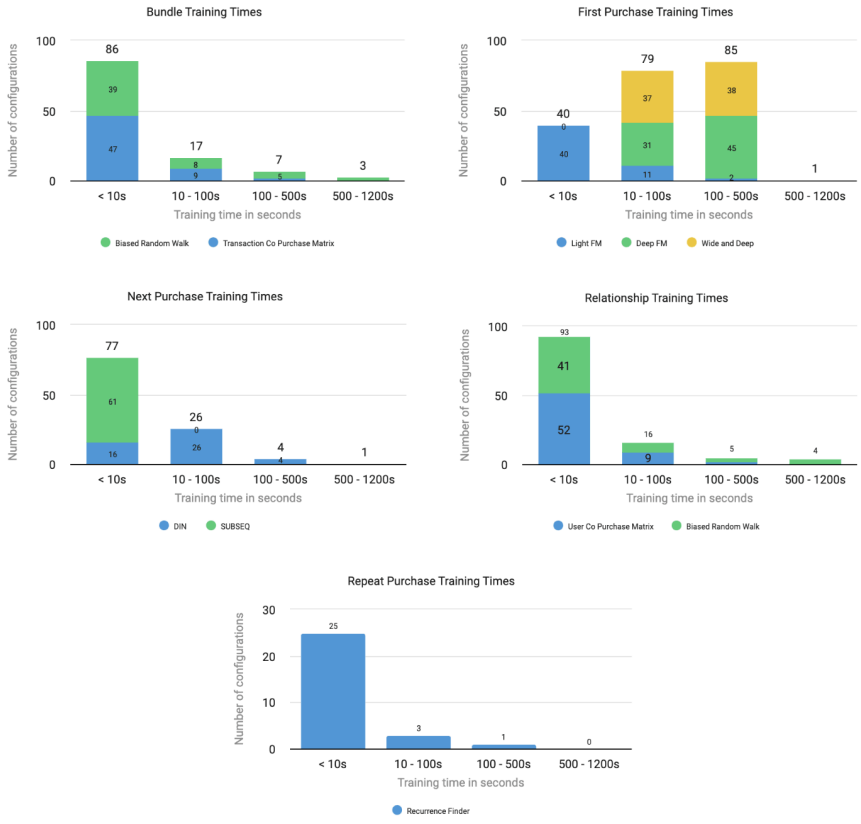

Fig. 2. Number of configuration vs training time across downstream tasks

Table 2. Dataset Overview

Metric	Count
Users	235
Items	64
Transactions	657
Purchasing users	235
Single purchase users	183
Multiple purchase users	52
User features	108

Table 3. Model training statistics for first purchase and sequence tasks

Task	Model	Accuracy	NDCG	Coverage
First Purchase	LightFM	0.34	0.34	0.05
First Purchase	DeepFM	0.04	0.03	0.03
First Purchase	Wide and Deep	0.02	0.01	0.03
Sequence	DIN	0.02	0.02	0.59
Sequence	SuBSeq	0.48	0.47	0.2

Table 4. Model training statistics for bundle and relationship tasks during training

Task	Model	Precision	Recall	NDCG	Coverage
Relationship	User Co Purchase	0.17	0.16	0.23	0.28
Relationship	Biased Random Walk	0.28	0.22	0.26	0.2
Bundle	Transaction Co Purchase	0.42	0.78	0.78	0.31
Bundle	Biased Random Walk	0.42	1	0.75	0.09

4 Conclusion

In this paper, we present our approach to solving existing challenges in creating a unified recommendation framework from a CRM software point of view. We discuss the challenges of serving a diverse customer base with varying amounts of transactional data. To cater to all transaction sizes, we train a catalogue of models from pattern mining to deep learning for each downstream task and dynamically select the best model by ranking them on a weighted combination of evaluation and scalability metrics. Each evaluation metric is run on a carefully

Table 5. Training scalability statistics

Model	Training Time (s)	Peak Memory (KB)
LightFM	0.61	2923.85
DeepFM	15.68	14317.79
Wide and Deep	15.93	13719.19
DIN	5.42	6546.41
SuBSeq	0.03	90.02
User Co Purchase	0.14	103.52
Biased Random Walk	0.68	769.57
Transaction Co Purchase	0.25	279.59
Biased Random Walk	0.66	803.3

designed evaluation dataset to ensure fair comparison across models. Furthermore, to aid our customers in their sales processes, we discuss our approach to provide reasonable explanations to our recommendation system using product and user-inverted indices. The recommendation reasoning coupled with our insights allowed our customers to improve lead conversion rates and other sales metrics.

Acknowledgments. A bold run-in heading in small font size at the end of the paper is used for general acknowledgments, for example: This study was funded by X (grant number Y).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Zoho CRM. https://www.zoho.com/crm, last accessed 2024/3/1
2. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate (2016)
3. Bhagat, R., Muralidharan, S., Lobzhanidze, A., Vishwanath, S.: Buy it again: Modeling repeat purchase recommendations. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. p. 62–70. KDD '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3219819.3219891, https://doi.org/10.1145/3219819.3219891
4. Cheng, H.T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., Shah, H.: Wide & deep learning for recommender systems. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. p. 7–10. DLRS 2016, Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2988450.2988454, https://doi.org/10.1145/2988450.2988454

5. Eksombatchai, C., Jindal, P., Liu, J.Z., Liu, Y., Sharma, R., Sugnet, C., Ulrich, M., Leskovec, J.: Pixie: A system for recommending 3+ billion items to 200+ million users in real-time. In: Proceedings of the 2018 World Wide Web Conference. p. 1775–1784. WWW '18, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, CHE (2018). https://doi.org/10.1145/3178876.3186183, https://doi.org/10.1145/3178876.3186183
6. Guo, H., Tang, R., Ye, Y., Li, Z., He, X.: Deepfm: a factorization-machine based neural network for ctr prediction. In: Proceedings of the 26th International Joint Conference on Artificial Intelligence. p. 1725–1731. IJCAI'17, AAAI Press (2017)
7. Hermenegildo Gil-Gomez, Vicente Guerola-Navarro, R.O.B., Lozano-Quilis, J.A.: Customer relationship management: digital transformation and sustainable business model innovation. Economic Research-Ekonomska Istraživanja **33**(1), 2733–2750 (2020). https://doi.org/10.1080/1331677X.2019.1676283, https://doi.org/10.1080/1331677X.2019.1676283
8. Ktistakis, R., Fournier-Viger, P., Puglisi, S.J., Raman, R.: Succinct bwt-based sequence prediction. In: Hartmann, S., Küng, J., Chakravarthy, S., Anderst-Kotsis, G., Tjoa, A.M., Khalil, I. (eds.) Database and Expert Systems Applications. pp. 91–101. Springer International Publishing, Cham (2019)
9. Kula, M.: Metadata embeddings for user and item cold-start recommendations. In: Bogers, T., Koolen, M. (eds.) Proceedings of the 2nd Workshop on New Trends on Content-Based Recommender Systems co-located with 9th ACM Conference on Recommender Systems (RecSys 2015), Vienna, Austria, September 16-20, 2015. CEUR Workshop Proceedings, vol. 1448, pp. 14–21. CEUR-WS.org (2015), http://ceur-ws.org/Vol-1448/paper4.pdf
10. Linden, G., Smith, B., York, J.: Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing **7**(1), 76–80 (2003). https://doi.org/10.1109/MIC.2003.1167344
11. Malkov, Y.A., Yashunin, D.A.: Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. IEEE Trans. Pattern Anal. Mach. Intell. **42**(4), 824–836 (apr 2020). https://doi.org/10.1109/TPAMI.2018.2889473, https://doi.org/10.1109/TPAMI.2018.2889473
12. Pugh, W.: Skip lists: a probabilistic alternative to balanced trees. Commun. ACM **33**(6), 668–676 (jun 1990). https://doi.org/10.1145/78973.78977, https://doi.org/10.1145/78973.78977
13. Rendle, S.: Factorization machines. In: 2010 IEEE International Conference on Data Mining. pp. 995–1000 (2010). https://doi.org/10.1109/ICDM.2010.127
14. Schafer, J.B., Konstan, J., Riedl, J.: Recommender systems in e-commerce. In: Proceedings of the 1st ACM Conference on Electronic Commerce. p. 158–166. EC '99, Association for Computing Machinery, New York, NY, USA (1999). https://doi.org/10.1145/336992.337035, https://doi.org/10.1145/336992.337035
15. Schmidhuber, J.: Deep learning in neural networks: An overview. Neural Networks **61**, 85–117 (Jan 2015). https://doi.org/10.1016/j.neunet.2014.09.003, http://dx.doi.org/10.1016/j.neunet.2014.09.003
16. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: Kgat: Knowledge graph attention network for recommendation. In: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. p. 950–958. KDD '19, Association for Computing Machinery, New York, NY, USA (2019). https://doi.org/10.1145/3292500.3330989, https://doi.org/10.1145/3292500.3330989
17. Wang, Y., Wang, L., Li, Y., He, D., Liu, T.Y.: A theoretical analysis of ndcg type ranking measures. In: Shalev-Shwartz, S., Steinwart, I. (eds.) Proceedings of the

26th Annual Conference on Learning Theory. Proceedings of Machine Learning Research, vol. 30, pp. 25–54. PMLR, Princeton, NJ, USA (Jun 2013), http://proceedings.mlr.press/v30/Wang13.html

18. Xia, F., Liu, J., Nie, H., Fu, Y., Wan, L., Kong, X.: Random walks: A review of algorithms and applications. IEEE Transactions on Emerging Topics in Computational Intelligence **4**(2), 95–107 (Apr 2020). https://doi.org/10.1109/tetci.2019.2952908, http://dx.doi.org/10.1109/TETCI.2019.2952908
19. Zhou, G., Zhu, X., Song, C., Fan, Y., Zhu, H., Ma, X., Yan, Y., Jin, J., Li, H., Gai, K.: Deep interest network for click-through rate prediction. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. p. 1059–1068. KDD '18, Association for Computing Machinery, New York, NY, USA (2018). https://doi.org/10.1145/3219819.3219823, https://doi.org/10.1145/3219819.3219823