Sequential Banking Products Recommendation and User Profiling in One Go

Alexandre Boulenger[*]
Genify[†]
Beijing, China
alex@genify.ai

Davide Liu[*]
Genify
Beijing, China
davide@genify.ai

George Philippe Farajalla
Genify
Beirut, Lebanon
george.farajalla@genify.ai

ABSTRACT

How can banks recommend relevant banking products such as debit, credit cards or term deposits, as well as learn a rich user representation for segmentation and user profiling, all via a single model? We present a sequence-to-item recommendation framework that uses a novel input data representation, accounting for the sequential and temporal context of both item ownership and user metadata, fed to a multi-head self-attentive encoder. We assess the performance of our model on the largest publicly available banking product recommendation dataset. Our model achieves 98.9% Precision@1 and 40.2% Precision@5, outperforming a state-of-the-art model as well as a common XGBoost-based baseline model tailored for this dataset and a system reportedly employed in industry for this task. Next, using the encoder embedding we obtain a continuous representation of users and their past product behavior. We demonstrate, in a case study, that this representation can be used for user segmentation and profiling, both critical to decision-making in organizations; for example, in designing and differentiating value propositions. The proposed approach is more inclusive and objective than the traditional ones employed by banks. With this work, we expose the benefits of employing a recommendation model based on self-attention in a real-world setting. The continuous user representation learned can yield far more impact than individual user-level recommendations. Both the proposed model and approach to segmentation and profiling are also applicable in other industries, beyond banking.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Neural networks**; • **Applied computing** → *Decision analysis.*

KEYWORDS

recommender system, sequence-to-item, self-attention, clustering, user profiling

[*]Both authors contributed equally to this research.

ACM Reference Format:
Alexandre Boulenger, Davide Liu, and George Philippe Farajalla. 2022. Sequential Banking Products Recommendation and User Profiling in One Go. In *3rd ACM International Conference on AI in Finance (ICAIF '22), November 2–4, 2022, New York, NY, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3533271.3561697

1 INTRODUCTION

The aim of a product recommender system is to recommend to users interesting products that could catch their attention and interest, thus leading them to interact and eventually purchase those items. To do so, recommendation engines can make use of user-relevant data to extrapolate meaningful information to capture user preferences and interests. Recommender systems have been prevalent since the end of the twentieth century when most applications were in e-commerce, driven by companies such as Amazon, eBay, and CDNow [16]. Nowadays, recommendation engines are still widely used in the e-commerce industry, and have gained popularity in other fields too following recent progress. For example, Alibaba [21] uses advanced algorithms to recommend new items to customers according to their past interactions and their current session, thus reducing the user's search scope and surprising them by proposing potentially attractive products.

The recent BERT model [7], based on the Transformer architecture [20], is considered a major innovation in neural machine translation and has been extended to other machine learning fields, leading to the development of recommender systems based on self-attention. What makes these model architectures particularly attractive is the continuous input embedding that is learned during training, which has repeatedly proven useful to complete downstream tasks distant from the original one, such as user segmentation and profiling.

These two tasks are of high importance to organizations managing a large userbase. Personalized offerings allow such organizations to improve their service quality and give their customers a better sense of satisfaction. This process usually involves first understanding the userbase and then segmenting it into different groups. Traditional segmentation methods involve accounting for past purchases via a bag-of-products or bag-of-categories approach, and user characteristics such as age or other demographic and economic metadata, but without taking into consideration the sequential and temporal contexts: the order and timing of the purchases–a shortcoming we address in the present paper.

The main contributions of our work are the following:

- We present a self-attention-based framework to perform sequence-to-item recommendation, via a novel input data

[†]https://www.genify.ai/

representation scheme for sequential item ownership history and any form of temporal context. *
- We illustrate the effectiveness of our framework on a popular and publicly available retail banking product recommendation dataset and outperform both widely-used baselines and a state-of-the-art benchmark.
- We conduct a case study in which we perform a complete user segmentation (clustering) and profiling analysis, with a high potential for impactful decisions if reproduced by banks. By doing so, we show our model learns a meaningful continuous user representation in the encoder's embedding space and we unlock applications beyond the recommendation task itself.
- We challenge this superior performance by analyzing ethical considerations when deploying a system recommending banking products.

2 RELATED WORK

2.1 Deep Learning and Recommender Systems

Deep learning has recently influenced the development of recommendation engines, in particular, those based on collaborative filtering. He et al. [9] developed a framework called Neural Collaborative Filtering. They showed that using deeper layers of neural networks increases recommendation performance. Cheng et al. [5] combined a generalized linear model with a feed-forward neural network to benefit from the linear model's memorization and the neural network's generalization capabilities. However, these methods do not take into account the sequential behavior of users; a user is more likely to buy a mouse if they own a computer.

Recurrent Neural network (RNN), Gated Recurrent Unit (GRU), or Long Short-Term Memory (LSTM) systems have been employed to model behavioral sequences. User-based RNNs and GRUs developed by Donkers et al. [8] outperform traditional benchmark recommender systems and other RNN-based systems. Experimental results indicate that the model learns implicit relations between events when using external user information, suggesting that the inherent dependency between successive events can be exploited.

More recent work exploits LSTM to produce recommendations [10] but is applied only in session-based contexts. Wu et al. [22] also use an LSTM model to generate recommendations but do not introduce any particular data representation scheme, while Tang and Wang [19] encode the input as an image, which is not directly applicable here.

2.2 Attention Mechanism in Recommender Systems

Given sequences of past user-item interactions, and user-specific context, the goal of a modern recommendation engine is to learn patterns from the temporal and contextual data to predict which items a user could use or buy in the future. Among possible approaches, the Transformer architecture [20] has proven able to yield superior performance at recommendation tasks.

Chen et al. [3], by using a Transformer-based system to model and capture sequential behavior, were able to improve online click-through-rate. Shao et al. [17] employed a Transformer for consecutive basket recommendation of student curriculum, using students' past course histories and future pre-specified courses of interest.

2.3 User Profiling

Learning user representations (also known as user profiling) by encoding both user behavior and characteristics, is a promising way of improving the quality of recommendations.

With the session-based recommender system DSAN, Yuan et al. [23] explored the interaction and correlation between each item within the session, learning a representation of the users' current preference based on the session information fed to a self-attention network. U-BERT [15] introduced a user encoder and a review encoder to generate a comprehensive user representation to use in downstream tasks for which training data may be limited. However, while they assess the quality of the learned representations by measuring recommendation performance quantitatively, none of them examine the meaning and usefulness of the learned representations for the problem at hand.

3 FRAMEWORK, INPUT REPRESENTATION AND MODEL

3.1 Framework

We define the products owned or used by a user as *products* and their personal information as *metadata*. We refer to the union of the *products* and *metadata* as a user's *data*. Consider a set of customers C, their corresponding metadata M and a set of products B. $B_c \subset B$ is the set of products owned by user $c \in C$ and M_c their metadata. We denote by $H_c = B_c \cup M_c$ the data of user c.

A user's metadata M_c consist of their demographic information, such as age, gender, income, place of residence, and other application-specific information such as seniority with a particular service provider, most of which vary over time. A user's products B_c correspond to the products owned or used over a specific time period. Hence, it is natural to view these as time-varying sequences of variables.

Given a sequence of products B_c^t and a sequence of metadata M_c^t for $t = [0, ..., T]$ where T is the length of the sequence, the goal of the model is to learn a function $f : (c, H_c^0, ..., H_c^T)$ that captures the sequential information to predict the probability of acquiring items in B in period $T + 1$. For simplicity, we call the sequence of data of a user as their *history*.

3.2 Sequential Input Data Representation

Given the similarity to a language modeling task, we represent the periods as words, and data of a single period as letters. The input to the model can be seen as a sentence containing T words of fixed number of letters each, corresponding to the history formed by T sequences of data values. In Equation (1) we show how a word of a sentence is represented, in the case of the present dataset.

$$[\ \underbrace{x_1}_{\text{year}}\ \underbrace{x_2 ... x_{13}}_{\text{month}}\ \underbrace{x_{14} ... x_{17}}_{\text{segmentation}}\ \underbrace{x_{18}}_{\text{age}}\ \underbrace{x_{19}}_{\text{income}}\ \underbrace{x_{20}}_{\text{seniority}}\ \underbrace{x_{21} ... x_{42}}_{\text{products}}\ ] \quad (1)$$

*https://github.com/genifyai/banking-reco-profile-icaif22

Our data representation yields the following advantages over sequences of actions:

- Both temporal and static metadata can be encoded.
- It can explicitly capture, at each timestamp, which items are used (owned) and which are not.
- Since we are in the presence of a small product space (< 25), a one-hot representation is suitable and does not result in lengthy sequences.
- Any time embedding can be encoded in such representation.

This flexible temporal input representation can be employed on problems and datasets with temporal meta and item data. It is not specific to, but particularly useful in recommendation problems with product universes limited to 100s but decade-long histories (sparse behavior), such as banking, insurance, real estate, or specialized e-commerce. In case a dataset presents a large number of products such that this one-hot-representation is inconvenient, a sub-set including the most recent items could be considered instead of the whole product space.

3.3 Model

Following the idea of BERT [7], we keep the Transformer architecture removing the decoder and only using the encoder. The encoder learns an embedding representation of the user input data, and then a feed-forward neural network applied on top of the output embedding predicts probabilities over the product set. The model outputs a probability $P_b >= 0$ for each product $b \in B$.

Each encoder layer is composed of a multi-head self-attention layer followed by a batch normalization layer, a feed-forward layer, and another batch-normalization layer. Each encoder layer is followed by a dropout layer with dropout probability of 0.5, to reduce overfitting. The encoder layers are followed by a feed-forward layer, itself followed by a softmax layer used to predict the probability of each item being purchased in the next period $T+1$. The architecture of the model is presented in Figure 1. To predict only the new items purchased, we mask from P_c^B the items that are already owned in the final period. More details about the hyperparameters of our model are shown in Table 1.

4 DATASET

4.1 Dataset Overview

We make use of the Santander Product Recommendation dataset, the largest publicly available dataset for retail banking products recommendation, used in a popular Kaggle competition [11]. With data spanning 17 months, the sequences start on 2015–01–28 and end on 2016–05–28. The data contains, for each user, information about which items they own each month among the 22 available, such as a credit card, savings account, etc., as well as personal information such as average income, age, or gender.

In total, records contain 48 monthly variables for each user. 22 of the said variables correspond to the products owned (products) and the other 26 variables are user characteristics (metadata). Among the metadata features, we only retained the four most important ones for subsequent study: `Age`, `Gross Household Income`, `Seniority` (with the bank), and `Segment`.

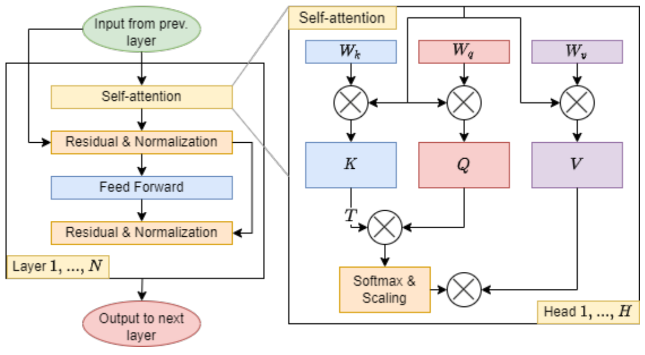

Figure 1: Diagram showing the model's multi-layer, multi-head self-attention encoder architecture, with details on a single self-attention head. In our case, there are $H = 7$ self-attention heads in each of the $N = 6$ encoder layers. Note that the last, non-repeating layers of the model (feed forward followed by softmax) are not shown here.

4.2 Data Preparation

The original dataset spans 17 months, from 2015–01–28 to 2016–05–28. The first 16 months are used for training, and the last month is used for testing. `Segment` is the only metadata variable one-hot encoded. The other three are numerical variables and are normalized to range within $[0, 1]$ using min-max normalization.

Sequence positional encoding is performed by attaching the binary representation of the year along with one-hot vector of length 12 representing the month of a record. For each user, we construct 16 sequences of length 42 corresponding to the months from 2015–01–28 to 2016–04–28. Within the sequences, we encode:

- The binary representation of the year when the data was sampled (0=2015, 1=2016).
- A one-hot vector of length 12 representing the month.
- 1 position for each of `Gross Household Income`, `Age`, and `Seniority`.
- A one-hot vector of length 4 representing `Segment`.
- A one-hot vector of length 22 representing products owned.

It is desirable to include the metadata at each time step since a change in metadata may reflect a change in user intention. For example, some high-income users who suddenly face an income drop may accordingly stop using particular banking products.

5 EXPERIMENTAL SETUP

We first compare the performance of our model against a benchmark and two baselines, all of which are available publicly. We show that our recommender system outperforms all benchmarks on the Santander dataset, across all metrics. We then analyze the model's sensitivity to input metadata, the seniority of the users with the bank, and the number of items owned.

5.1 Baselines

Amazon Personalize (AP) is an auto-machine learning service that employs a Hierarchical Recurrent Neural Network (HRNN) [1] to generate recommendations. It takes ordered user product history and weights it to perform better inference. We include this baseline for comparison as it is known to be employed in industry specifically on the task of banking products recommendation [12].

XGBoost is an ensemble machine learning algorithm that uses a gradient boosting framework on decision trees [4]. The XGBoost baseline is preceded by a specific data processing scheme crafted specifically for the Santander dataset [11]. The multiple binary dependent variables are collapsed into a single, multi-value dependent variable and if a user acquired multiple products in a given month, it is repeated once for each product. We include this model as a baseline for comparison as it is the highest-performing open-sourced model [2] in the leaderboard of the Santander Product Recommendation competition [11] and has been previously used as a baseline to product recommender systems [13].

BERT4Rec [18] employs deep bidirectional self-attention to model user behavior sequences. To avoid information leakage and efficiently train the bidirectional model, it adopts the Cloze objective [7]: it predicts randomly masked items in the sequence by jointly conditioning on left and right contexts. With this feature, the model learns a bidirectional representation to generate recommendations by allowing each "word" to fuse information from both sides. We consider this benchmark as it is a state-of-the-art model with an architecture similar to that of our model.

5.2 Metrics

To measure our model's performance we employ Prec@k (often known as Precision@k) and Rec@k (Recall@k) with $k = 1, 5, 10$, Mean Reciprocal Rank (MRR) and Normalized Discounted Cumulative Gain (NDCG) metrics [6].

5.3 Training

The model learns using Adam to optimize a binary cross entropy loss function given the recommended and ground truth acquired products. To fine-tune the learning rate, we adopt a RAdam warm-up schedule with a peak after 5 epochs and decay until the 20th epoch, when training terminates. By doing so we have a higher learning rate initially when the parameters of the model are still far from the optimum, and then gradually decrease it to provide a more stable training as the model converges. The initial value of the learning rate is fixed to 10^{-4} during training, and we adopt a batch size of $k = 32$ users. Hence, the encoder receives inputs of shape $[k, dim(T), dim(H)]$ and its output embedding space has shape $[k, dim(D)]$. The final probability distribution has shape $[k, dim(B)]$.

In Table 1, we list the hyperparameters that we tuned to optimize the performance, and for each, we show the values tried and the value selected using cross-validation. We tuned the number of warm-up epochs instead of the initial learning rate (set to 10^{-4}).

All experiments were run in Python using NumPy version 1.18.5, PyTorch version 1.8.1, and a GeForce GTX TITAN X GPU with 12212 MB memory. For the XGBoost-based baseline, we retain the hyperparameters employed by its authors on this dataset, and BERT4Rec is fine-tuned using a similar scheme as the one adopted to fine-tune our model. We set the batch size to 32, the maximum window length to capture the 70 most recent users' interactions, and train for 20 epochs. For AP, the end-to-end training of the model and tuning of hyperparameters do not require human assistance but are instead handled by an AutoML algorithm.

Table 1: Hyperparameter values tried and selected for our model.

	Values Tried	Selected
Encoder Layers	3, 5, 6, 7, 9, 10	6
Attention Heads	5, 6, 7, 8	7
Hidden Layer Size	256, 512, 1024, 2048	2048
Warm-up Epochs	3, 4, 5	5
Batch Size	32, 64, 128	32

6 RESULTS AND ANALYSIS

6.1 Performance Comparison

The performance of our model on the test set (17th month) and the baselines is reported in Table 2. Our model, better equipped to capture temporal dependencies and interactions between items of a sequence, significantly outperforms both the XGBoost and AP baselines across all metrics. Over 98% of the recommendations made by our model are products that have effectively been acquired. In comparison, less than 70% of the top-1 recommendations made by XGBoost are correct, and this value falls to less than 50% in the case of AP. Further, our model also outperforms BERT4Rec, albeit with a lower margin for improvement.

One component that contributes to the superior performance of our model is the input representation employed. By design, it is more suitable to deal with a relatively low number of timestamps, each possibly associated with multiple items, and allows for temporal or static metadata as additional input. Thanks to this data representation scheme, our model can take advantage of temporal user metadata, resulting in a enhanced performance, with metrics Prec@1 and Rec@1 benefiting most from it, while baseline models cannot make use of metadata by design. Even in the case when the model only receives the product history as input, thus excluding metadata, it still outperforms the baselines in the same way.

Prec@k and Rec@k decrease for larger values of k because for most examples in the dataset, ground-truth recommendations (acquired products) are few. That is, for k large enough, some recommendations are always wrong. If, for example, $k = 5$ but we only have three ground-truth items, Prec@5 will be at most 3/5 but Prec@1 can be 1. This is specific to the dataset and the problem on hand (where few banking products are acquired every year). Note that all models have been evaluated with the same metrics.

Although our main task is to predict item acquisition, we also report in Table 2 the performance on the task of predicting item ownership, that is, acquisition and retention (or continuation) together. The difference between acquisition and ownership is that the former does not include items owned in the last month T before the prediction month $T + 1$. Hence, items acquired in $T + 1$, are computed excluding the items in T from the original predictions. The same approach is followed for all benchmark models for fair comparison (note, the XGBoost-based baseline only predicts product acquisition by design). The superior performance of our model on the item acquisition task also holds on the item ownership task.

Table 2: Performance of our model compared to benchmark models on the item acquisition task (top), and item ownership task (bottom). We also report the performance of our model on both tasks when using only the product history (no metadata) as input. The XGBoost-based model only predicts product acquisition by design, so its performance is not reported on the item ownership task.

Task	Model	Prec@1	Prec@5	Prec@10	Rec@1	Rec@5	Rec@10	MRR	NDCG
Acquire	BERT4Rec	0.9693	0.3881	0.2125	0.6823	0.9581	0.9912	0.9830	0.9796
	Amazon Personalize	0.4653	0.1491	0.0935	0.4183	0.6396	0.7869	0.5788	0.6505
	XGBoost	0.6887	0.2449	0.1285	0.6062	0.9440	0.9866	0.8054	0.8556
	Our model	**0.9891**	**0.4022**	**0.2157**	**0.6975**	**0.9764**	**0.9979**	**0.9937**	**0.9941**
	Ours (no metadata)	0.9813	0.3995	**0.2157**	0.6911	0.9726	**0.9979**	0.9891	0.9895
Own	BERT4Rec	0.9812	0.3452	0.1852	0.7488	0.9736	0.9947	0.9901	0.9873
	Amazon Personalize	0.9622	0.2825	0.1664	0.7397	0.8865	0.9571	0.9788	0.9435
	Our model	**0.9920**	**0.3537**	**0.1875**	**0.7560**	**0.9836**	**0.9990**	**0.9956**	**0.9961**
	Ours (no metadata)	0.9833	0.3514	**0.1875**	0.7501	0.9805	0.9989	0.9907	0.9915

6.2 Sensitivity Analysis

When excluding metadata, most of the metrics slightly decrease, but the model still outperforms the benchmarks, as shown in Table 2. We believe that the limited gain attributed to the addition of the metadata is because temporal product ownership data are enough to learn user patterns. However, metadata are still useful to improve recommendations when the model can reveal a particular extrinsic behavior. For instance, a pension is more suitable for a customer with an advanced age while a student loan could be more suitable for customers whose segmentation falls in the *student* category.

Next, we investigate in Figure 2 and Figure 3 the variation in performance as the seniority of users with the bank and the number of products owned vary, respectively.

Prec@k, $k = 3, 5, 10$, increase with user seniority and peak slightly before seniority crosses the 100 months line, and beyond this they all drop. As users build a relationship with the bank, they settle into a certain behavior and basket of products and are unlikely to depart from it. Any change to items owned will come as unexpected and abrupt, making it harder for the model to predict the next acquisition beyond certain seniority. This finding can guide banks to decide when to employ the recommender system in a user's lifetime, and when not to.

In Figure 3, we observe a steady drop in precision as the number of items owned increases. As users acquire more items, their behavior profile becomes more specific and less likely to be shared by other users, hence harder for the model to learn.

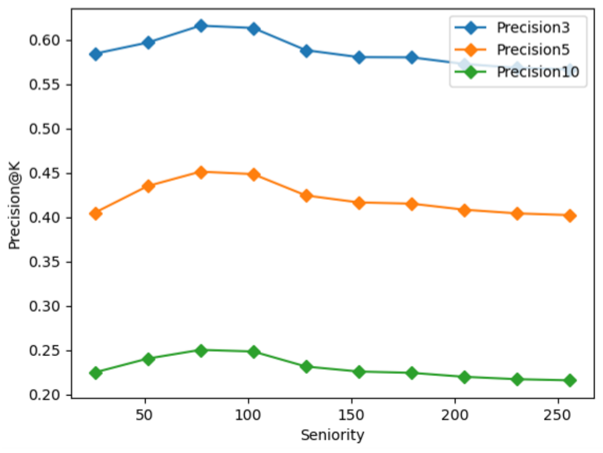

Figure 2: Sensitivity of Prec@k, $k = 3, 5, 10$ to user seniority (as client of the bank).

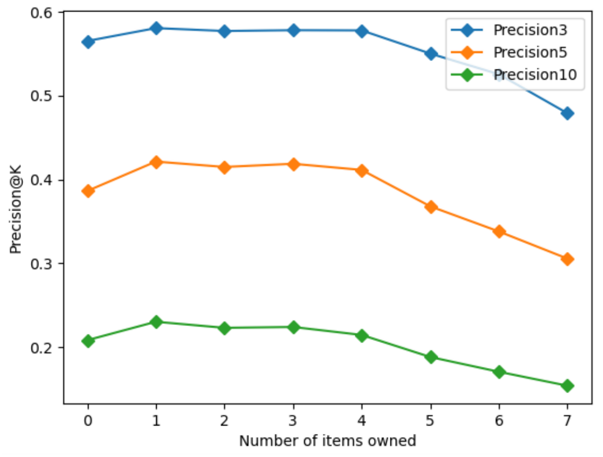

Figure 3: Sensitivity of Prec@k, $k = 3, 5, 10$ to the number of items owned.

7 CASE STUDY: USER PROFILING IN ENCODER EMBEDDING SPACE

By combining user metadata and product history into a single sequential input representation, and training our model to predict future purchases from all user parameters, past purchases, and their temporal context, we generate a meaningful continuous user representation in the encoder embedding space. We conduct a case study to exhibit how such a representation can be used by banks for user segmentation and profiling.

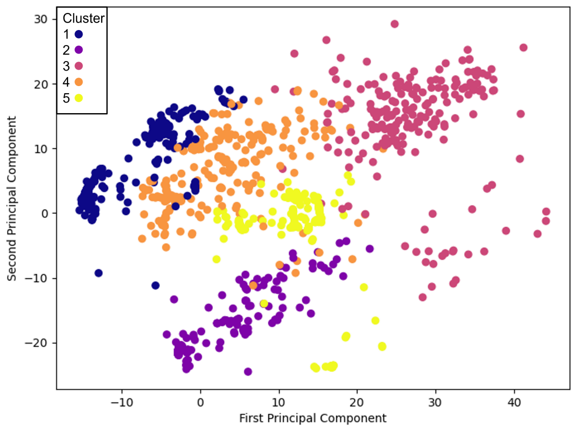

Figure 4: Scatter plot of the first two principal components of the encoder embedding. Each dot is a user in the dataset, colored according to the cluster they belong to.

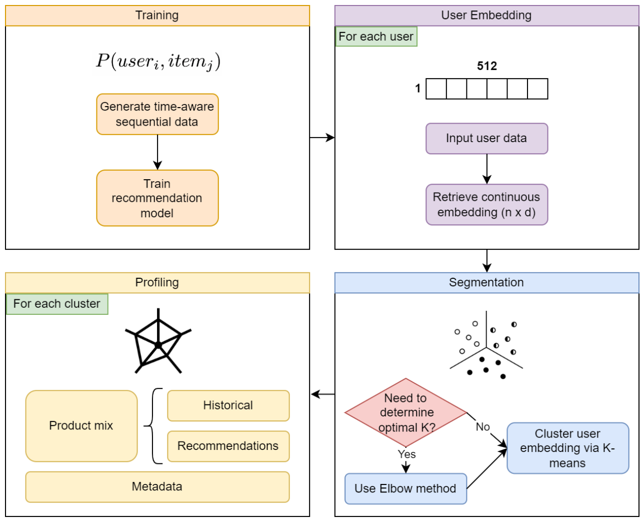

Figure 5: Diagram illustrating the end-to-end approach followed to segment and profile the users in the encoder embedding space.

7.1 Approach

Our approach to segmentation and profiling is described in the diagram in Figure 5. We first apply the unsupervised K-means algorithm to the encoder embedding. We opt to partition the embedding space into five clusters. The optimal number of clusters has been chosen according to the Elbow method, which compares the value of inertia of different numbers of clusters. Next, we visualize in Figure 4 the first two principal components of the embedding space, obtained via principal component analysis (PCA), and observe that the five clusters are also separable according to these.

7.2 Clusters and Profiling

We profile the clusters by analyzing their average past product mix and demographics in Table 3. Items owned by or recommended to 11% or fewer individuals in the dataset were excluded. The clusters have similar average demographics (except for Cluster 3) and core product mix ("Current account" is heavily present across Clusters 1, 2, 3, and 4). However, they differ widely with respect to their marginal product mix.

We also assess the relevance of the clusters by analyzing their (future) product mix as recommended by the model. "Credit cards", "Direct debit" and "Long-term deposits" are common recommendations across all five clusters, but each cluster has specific recommendations which differentiate it from the others. This shows our model can recommend relevant low-incidence products based on specific events in past product purchases.

In Figure 6, we offer a visual comparison of the clusters described in Table 3 in radar charts showing the distribution of past and recommended products in each of the clusters.

Regarding the historical product mix, we note clear differences between clusters, either in the presence of products (e.g., Cluster 5 is the only cluster with *Payroll* and *Pensions 2* products, Clusters 4 and 5 the only ones with the *Taxes* product) or in the magnitude of the penetration (distribution) of products in clusters (e.g., all clusters have *Current Accounts* and three clusters have *Particular Accounts*, but with varying degrees of penetration).

Similar variations between clusters can be observed in the distribution of recommended products. *Direct Debit* and *Credit Card* are recommended across all clusters but in varying degrees (25%-95+% and 30%-95+%, respectively). Each cluster has a distinct shape on the spider charts, denoting emphasis on a set of products (Clusters 4 and 5) or a single product (e.g., *Current Accounts* for Cluster 2, *Long-term Deposits* for Clusters 1 and 3). However, we also note some similarities between clusters; for example Clusters 3 and 5 are recommended *Direct Debit*, *Credit Card* and *Long-term Deposits* in similar proportions; these two clusters are also the clusters with the lowest average age.

Finally, with these spider charts, we are able to verify the recommendation coherence and sanity. Cluster 5 has more than 80% penetration of *Direct Debit* and is the cluster whose users are the least frequently recommended *Direct Debit* (20%). Cluster 2 has the lowest penetration of *Current Accounts*, below 20%, and receives *Current Accounts* recommendations most frequently (90%).

From the continuous user representation learned by the model, we proposed an approach to segment users according to high-level features most relevant to their future purchases, and subsequently profile the resulting segments. Banks can in turn differentiate between user groups similar on the sole basis of age or other characteristics. This case study highlights the added benefits of using encoder-based models for products recommendation.

8 ETHICAL CONSIDERATIONS

In this section, we analyze ethical considerations that arise when deploying a system recommending banking products and performing segmentation and profiling as presented in Section 7.

The segmentation and profiling approach underlying this case study is more inclusive, objective, and attractive than traditional

Table 3: Profiling of the user clusters according to their metadata, historical product ownership, and recommendations made by the model. Age is the average age in years. In the Product History and Recommendations columns, we detail the proportion of users in a given cluster that share common ownership or recommendation behavior, respectively.

#	Size	Age	Product History (*own or did own*)	Recommendations (*were recommended*)
1	10%	54	100% both current and particular accounts	80%-90% credit cards, direct debit, or, long-term deposit accounts
2	13%	48	11%-35% particular, current, junior and e-accounts	80%-90% direct debit or current accounts, 50% payroll
3	54%	33	99% current accounts, 10% direct debit	75%-98% credit cards, direct debit, or long-term deposits, 21% payroll
4	12%	49	99% current accounts, 45% e-accounts, 41% direct debit	60%-70% c. cards or d. debit, 25% long-term deposits, payroll, or pensions
5	11%	44	99% payroll accounts, 90% d. debit and pensions	23%-51% c. cards, current accounts, payroll, direct debit, mortgage, or pensions

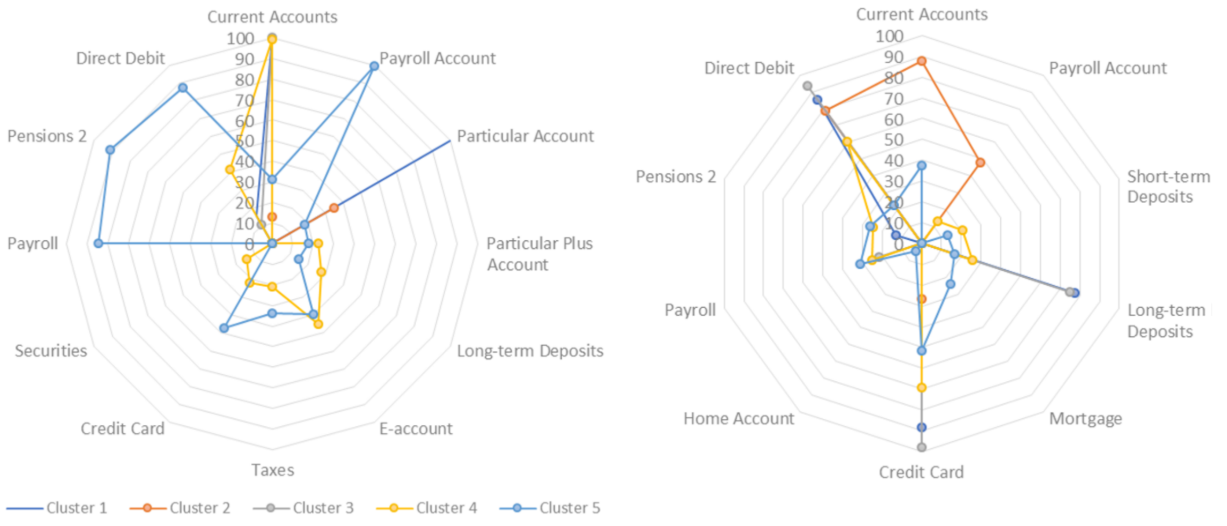

Figure 6: Historical (left) and recommended (right) product mix for each of the five clusters. Each color corresponds to a cluster, and each dot indicates, for a given cluster, the share of individuals in this cluster who own this product (*i.e.*, the penetration).

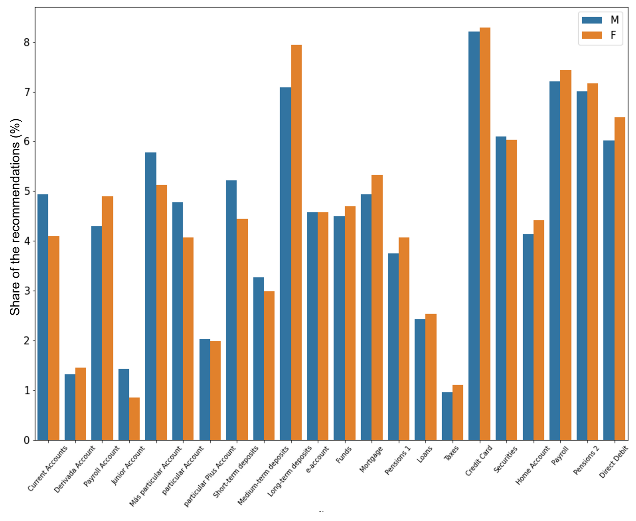

Figure 7: Distribution of the recommendations across items for each gender, male (M) and female (F).

approaches employed by banks. The embedding space in which we perform the segmentation is learned to optimize the relevance of the products recommended for future purchase. Hence, our segmentation and profiling is primarily driven by future purchase behavior–an attractive property when using the segmentation and profiling in a key downstream task such as the design of differentiated value propositions. Banks traditionally segment their userbase using solely demographic and economic characteristics such as age, gender, or income (all prone to discrimination), and at best also consider past user purchases in a bag-of-products fashion. Neither the temporal context, the order and timing of purchases nor future purchase behavior are taken into consideration. Our approach is hence not only more attractive, but also more inclusive by design.

To further analyze ethical considerations about our model and the problem it solves, we refer to the framework from Milano et al. [14] and discuss two areas of concern, *fairness*, and *content*. One can observe in Figure 7 that the recommendations are not gender-biased, that is, the distribution of recommendations across products is similar for both genders; for any product, the probability varies between male and female by at most 15%. This partially addresses the concern of *fairness*.

There is also an inherent risk of financial harm arising from banking products, whether a recommendation engine is involved or not (concern of *content*). We posit that the greater financial awareness brought about by exposure to banking product recommendations outweighs the risk of financial harm to the user. This is especially true as banking products in most cases cannot be acquired with the click of a button, and require further efforts by the user to apply and a clerk to review. The presence of disclaimers around the recommendations can help further mitigate this risk.

9 CONCLUSIONS AND FUTURE WORK

In this work, we model the recommendation problem as a sequence-to-item task. We introduce a framework based on self-attention and a novel representation scheme, to perform item recommendations from any type of sequential input data and temporal user context. Drawing an analogy with language models, we represent, for each user, owned products and user metadata as "letters", and time periods as "words", to form "sentences" that account for the temporal context of item ownership and user metadata.

Our model significantly outperforms strong and widely-used benchmarks, highlighting the effectiveness of the attention mechanism on tasks involving sequential input data. This benchmarking was conducted on the public retail banking product recommendation dataset from Santander Bank [11].

Out of all inputs, we observed that, besides the product history, temporal user metadata are instrumental to recommendation performance. While the ability to use metadata with temporal context is what sets our model apart, even when deprived of metadata as input, our model still outperforms the baselines.

The trained model yields, from its encoder embedding, a continuous user representation, which can be used to perform user segmentation and profiling, or other high-value downstream applications, in retail banking and other industries. In a case study, we showed how a bank can segment its userbase and generate meaningful clusters that differ widely with respect to their average historic product mix and recommended product mix.

This is a highly desirable by-product for banks, and other user-centered or commercial organizations alike. Indeed, they typically conduct segmentation exercises to boost sales with differentiated value propositions. However, they too often limit themselves to segments defined according to metadata, or, at best, static product ownership. The segmentation approach we propose not only accounts for the entire sequential and temporal contexts of user characteristics and product ownership but is performed on higher-level features learned to predict future purchases–precisely what such organizations aim to drive up. The explainability brought about in the process is also key to ensuring the adoption of the recommender system itself.

REFERENCES

[1] AWS. 2021. Amazon Personalize Hierarchical Recurrent Neural Network. https://docs.aws.amazon.com/personalize/latest/dg/native-recipe-hrnn.html. Accessed: 2021-05-03.

[2] BreakfastPirate, Kaggle. 2021. XGBoost Solution of the Santander Product Recommendation Kaggle Competition. https://www.kaggle.com/c/santander-product-recommendation/discussion/25579. Accessed: 2021-05-03.

[3] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. 2019. Behavior Sequence Transformer for E-commerce Recommendation in Alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*. 1–4.

[4] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (San Francisco, California, USA) *(KDD '16)*. ACM, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

[5] Heng-Tze Cheng, Levent Koc, Jeremiah Harmsen, Tal Shaked, Tushar Chandra, Hrishi Aradhye, Glen Anderson, Greg Corrado, Wei Chai, Mustafa Ispir, et al. 2016. Wide & Deep Learning for Recommender Systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*. 7–10.

[6] cran. 2022. Evaluating recommender systems. https://cran.r-project.org/web/packages/recometrics/vignettes/Evaluating_recommender_systems.html. Accessed: 2022-07-03.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[8] Tim Donkers, Benedikt Loepp, and Jürgen Ziegler. 2017. Sequential User-based Recurrent Neural Network Recommendations. In *Proceedings of the 11th ACM Conference on Recommender Systems*. 152–160.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural Collaborative Filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[10] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. arXiv:1511.06939 [cs.LG]

[11] Kaggle. 2021. Santander Product Recommendation Kaggle Competition. https://www.kaggle.com/c/santander-product-recommendation/. Accessed: 2021-05-03.

[12] Karl Flinders. 2021. Emirates NBD builds bank of future with Amazon Web Services. https://www.computerweekly.com/news/252464527/Emirates-NBD-builds-bank-of-future-with-Amazon-Web-Services. Accessed: 2021-05-03.

[13] Chieh Lo, Hongliang Yu, Xin Yin, Krutika Shetty, Changchen He, Kathy Hu, Justin M Platz, Adam Ilardi, and Sriganesh Madhvanath. 2021. Page-level Optimization of e-Commerce Item Recommendations. In *Fifteenth ACM Conference on Recommender Systems*. 495–504.

[14] Silvia Milano, Mariarosaria Taddeo, and Luciano Floridi. 2020. Recommender systems and their ethical challenges. *AI & SOCIETY* 35, 4 (2020), 957–967.

[15] Zhaopeng Qiu, Xian Wu, Jingyue Gao, and Wei Fan. 2021. U-BERT: Pre-training User Representations for Improved Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (May 2021), 4320–4327. https://ojs.aaai.org/index.php/AAAI/article/view/16557

[16] J Ben Schafer, Joseph Konstan, and John Riedl. 1999. Recommender Systems in E-commerce. In *Proceedings of the 1st ACM Conference on Electronic Commerce*. 158–166.

[17] Erzhuo Shao, Shiyuan Guo, and Zachary A Pardos. 2021. Degree Planning with PLAN-BERT: Multi-Semester Recommendation Using Future Courses of Interest. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35(17). 14920–14929.

[18] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 1441–1450.

[19] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 565–573.

[20] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is All You Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[21] Jizhe Wang, Pipei Huang, Huan Zhao, Zhibo Zhang, Binqiang Zhao, and Dik Lun Lee. 2018. Billion-scale Commodity Embedding for E-commerce Recommendation in Alibaba. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 839–848.

[22] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J. Smola, and How Jing. 2017. Recurrent Recommender Networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining* (Cambridge, United Kingdom) *(WSDM '17)*. Association for Computing Machinery, New York, NY, USA, 495–503.

[23] Jiahao Yuan, Zihan Song, Mingyou Sun, Xiaoling Wang, and Wayne Xin Zhao. 2021. Dual Sparse Attention Network For Session-based Recommendation. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (May 2021), 4635–4643. https://ojs.aaai.org/index.php/AAAI/article/view/16593