Next Basket Prediction using Recurring Sequential Patterns

Riccardo Guidotti[*1,2], Giulio Rossetti[†1,2], Luca Pappalardo[‡1,2], Fosca Giannotti[§2], and Dino Pedreschi[¶1]

[1]Department of Computer Science, University of Pisa, Italy
[2]ISTI-CNR, Pisa, Italy

February 24, 2017

Abstract

Nowadays, a hot challenge for supermarket chains is to offer personalized services for their customers. *Next basket prediction*, i.e., supplying the customer a shopping list for the next purchase according to her current needs, is one of these services. Current approaches are not capable to capture at the same time the different factors influencing the customer's decision process: co-occurrency, sequentuality, periodicity and recurrency of the purchased items. To this aim, we define a pattern *Temporal Annotated Recurring Sequence* (*TARS*) able to capture simultaneously and adaptively all these factors. We define the method to extract TARS and develop a predictor for next basket named *TBP* (*TARS Based Predictor*) that, on top of TARS, is able to to understand the level of the customer's stocks and recommend the set of most necessary items. By adopting the TBP the supermarket chains could crop tailored suggestions for each individual customer which in turn could effectively speed up their shopping sessions. A deep experimentation shows that TARS are able to explain the customer purchase behavior, and that TBP outperforms the state-of-the-art competitors.

[*]riccardo.guidotti@di.unipi.it
[†]giulio.rossetti@di.unipi.it
[‡]luca.pappalardo@di.unipi.it
[§]fosca.giannotti@isti.cnr.it
[¶]dino.pedreschi@di.unipi.it

1 Introduction

Detecting the purchase habits of customers and their evolution in time is a crucial challenge for effective marketing policies and engagement strategies. In such context one of the most promising facilities retail markets can offer to their customers is *next basket prediction*, i.e., the automated forecasting of the next basket that a customer will purchase. Indeed, an effective basket recommender can act as a *shopping list reminder* for a customer, suggesting the items that she could probably need.

A successful realization of this application requires an in-depth knowledge of an individual's general and recent behavior [20]. In fact, purchasing patterns of individuals evolve in time and can experience deep changes due to both environmental reasons, like seasonality of products or retail policies, and personal reasons, like diet changes or shift in personal status or preferences. As consequence, a satisfactory solution to next basket prediction must be highly *adaptive* to the evolution of a customer's behavior, the recurrence of her purchase patterns and their periodic changes.

In this paper we propose the *Temporal Annotated Recurring Sequences* (*TARS*), adaptive patterns which model the purchasing behavior of an individual by four main characteristics. Firstly TARS consider the *co-occurency*: a customer systematically purchases a set of items together. Secondly TARS model the *sequentiality* of purchases, i.e., the fact that a customer systematically purchases a set of items after another one. Third TARS consider *periodicity*: a customer can systematically make a sequential purchase only in specific periods of the year, because of environmental factors or personal reasons. Fourth, TARS consider the *recurrency* of a sequential purchase during each period, i.e., how frequently that sequential purchase appears during a customer's period of the year. Modeling these four aspects – co-occurence, sequentiality, periodicity and recurrency – is fundamental in our opinion to detect the behavior of an individual and its evolution in time. On one hand future needs depend on the needs already satisfied: what a customer will purchase depends on what she already purchased last time. On the other hand, the needs of a customer depends on her specific habits, i.e., recurring purchases she makes over and over again. However habits are far from being static, since they are affected by both endogenous and personal factors [5, 17]. Therefore, periodicity is a crucial characteristic of an adaptive model for next basket prediction. We exploit the TARS and the multiple factors they are able to capture for constructing a parameter-free *TARS Based Predictor* (*TBP*). TBP is able to solve the next basket prediction problem and to provide a reliable list of items to be reminded in the next purchase as basket recommendation.

We demonstrate the effectiveness of our approach by extracting the TARS for thousands of customers in three real-world datasets, including unique datasets covering a seven years long period. We show how TARS are easily readable and interpretable, a characteristic which allows to gain useful insights about the purchasing patterns of products and customers. Then, we implement a repertoire of state-of-the-art methods and compare them with TBP. Our results show that *(i)* TBP outperforms the competitors, *(ii)* it is able to predict up to the next 20 baskets, and *(iii)* the quality of its predictions stabilizes after about 36 weeks.

Finally, it is worth underlining that both TARS and TBP are markedly *user-centric* approaches, in the sense that they use just the data of a customer to make predictions about that customer [22, 12]. This aspect eases the customers' personal data management and allows for developing tailored basket recommenders that can even run on the customers' mobile devices [6, 29].

In summary, our contributions are the following: *(i)* we introduce TARS and a parameter-free algorithm to extract them (Section 4); *(ii)* we develop TBP, a predictor based on TARS able to produce a shopping list reminder (Section 5); *(iii)* we extract TARS from real-world datasets and show how they are easily interpretable and readable (Section 6); *(iv)* we characterize TBP and compare it with state-of-the-art methods on real datasets (Section 6). The rest of the paper is organized as follows. Section 2 reviews existing approaches and Section 3 formalizes the problem. Finally, Section 7 concludes the paper suggesting future research directions.

2 Related Work

Next basket prediction is mainly aimed at the construction of effective recommender systems (or recommenders). Recommenders can be categorized into *general*, *sequential*, *pattern-based* and *hybrid*. General recommenders are based on collaborative filtering and produce recommendations with respect to general customers' preferences [27]. Sequential recommenders are based on Markov chains and produce recommendations exploiting sequential information and recent purchases [3]. Pattern-based recommenders base predictions on frequent itemsets extracted from the purchase history of all customers while discarding sequential information [11, 18].

The hybrid approaches combine the ideas underlying general and sequential recommenders. In [24] the authors use personalized transition graphs over Markov chains and compute the probability that a customer will purchase an item by using an optimization criteria named Bayesian Personalized Ranking [23]. HRM [31] and DREAM [32] exploit both general customers' preferences and sequential information by using recurrent neural networks. A different hybrid approach is described in [30] where is developed a probability model by merging Markov chain and association patterns.

All the approaches described above suffer from several limitations. General recommenders and pattern-based recommenders do not take into account neither the sequential information (i.e., which item is bought after which) nor the customers' recency. On the other hand, sequential recommenders assume the independence of items in the same basket and do not capture factors like mutual influence. Furthermore, all of them require transactional data about many customers in order to make a prediction for a single customer. For this reason, they do not follow the *user-centric* vision for data protection as promoted by the World Economic Forum [12, 22], which incentives personal data management for every single user of a data-based service. Cumby et al. [4] propose a basket predictor which embraces the user-centric vision by reformulating next basket prediction as a classification problem: they build a distinct classifier for every customer and for every item hence performing predictions by relying just on her personal data. However, this approach also assumes the independence of items purchased together.

Finally, the main drawback of the existing hybrid approaches [32, 31, 30] is that their predictive models are hardly readable and interpretable by humans. Interpretability of a predictive model, i.e., the possibility to understand the mechanisms underlying the predictions [25], is highly valuable for a retail chain manager interested in interpreting the predictive model to improve the marketing strategies and the service offered. Moreover, interpretability is also important to the customers who want to gain insights about their personal purchasing behavior.

3 Next Basket Prediction Problem

We refer to *next basket prediction* as the task of predicting which items a customer will purchase in her next transaction. Formally, let $C = \{c_1, \dots, c_z\}$ be a set of z customers and $I = \{i_1, \dots, i_m\}$ be a set of m items. Given a customer c, $B_c = \langle b_{t_1}, b_{t_2}, \dots, b_{t_n} \rangle$ is the ordered *purchase history* of her baskets (or transactions), where $b_{t_i} \subseteq I$ represents the basket composition and $t_i \in [t_1, t_n]$ is the transaction time. We indicate with $\mathcal{B} = \{B_{c_1}, B_{c_2}, \dots, B_{c_z}\}$ is the set of all customers' purchase histories.

Given the purchase history B_c of customer c and the time t_{n+1} of the next transaction, next basket prediction consists in providing the set b^* of k items that c will purchase in the next transaction $b_{t_{n+1}}$.

Our approach to next basket prediction aims at overcoming the main limitations of existing methods illustrated in Section 2. To this purpose, we propose a hybrid predictor which combines ideas underlying sequential and pattern-based recommenders. The approach consists of two main components. The first one is the extraction of *Temporal Annotated Recurring Sequences* (*TARS*) from the customer's purchase history, i.e., sequential recurring patterns able to capture the customer's purchasing habits. The second one is in the *TARS Based Predictor* (*TBP*), a predictive method that exploits the TARS of a customer to forecast her next basket.

4 Capturing Purchasing Habits

In this section we formalize TARS and describe how to extract them form the purchase history of a customer.

4.1 Temporal Annotated Recurring Sequences

Temporal Annotated Recurring Sequences (*TARS*) model recurrent sequential purchases of a customer, i.e., the fact that a set of items are typically purchased together, the fact that that a set of items is typically purchase after another set of items, and the recurrence of the sequential purchase, i.e., when and how often it occurs in the purchase history of the customer. We define a TARS as follows:

Definition 1 (Temporal Annotated Recurring Sequence) *Given the purchase history B of a customer, a* temporally annotated recurring sequence *(*TARS*) is a quadruple $\gamma=(S,\alpha,p,q)$, where $S=\langle X,Y\rangle=X\rightarrow Y$ is the* sequence *of itemsets, $\alpha=(\alpha_1,\alpha_2)\in\mathbb{R}^2_+$, $\alpha_1\leq\alpha_2$ is the* temporal annotation*, p is the* number of periods *in which the sequence recurs, and q is the* median of the number of occurrences in each period*. A TARS will also be represented as follows:*

$$\gamma = X \xrightarrow[p,q]{\alpha} Y$$

We refer to $\Gamma_c=\{\gamma_1,\dots,\gamma_m\}$ as the set of all the TARS of a customer c. A TARS is based on the concept of *sequence*, $S=\langle X,Y\rangle=X\rightarrow Y$, which intuitively indicates that the an itemset Y is typically purchased after another itemset X. The itemsets themselves point out which items are purchased together. For example, a sequence $\{a\} \to \{b, c\}$ indicates that the items $\{b, c\}$ are purchased together after the itemset $\{a\}$. The temporal annotation $\alpha = (\alpha_1, \alpha_2)$ indicates the minimum intra-time α_1 and maximum intra-time α_2 *intra-time* of the sequence, i.e., the range of time elapsing between the purchase of X and the purchase of Y. A sequence can appear in several distinct *periods*, i.e., time intervals where the sequence occurs continuously. The number of periods p characterizes these recurrences, that is, in how many periods the sequence S appears. Finally, q indicates how many times S typically occurs in a period.

Definition 2 (Sequence) *Given the purchase history of a customer* $B_c = \langle b_{t_1}, \dots, b_{t_n} \rangle$*, we call* $S = \langle X, Y \rangle = X \to Y$ *a* sequence *if the pair of itemsets* $X \subseteq b_{t_h}$ *and* $Y \subseteq b_{t_l}$*,* $X, Y \neq \emptyset$*,* $t_h < t_l$ *and* $\nexists\, S' = X' \to Y'$*,* $X' \subseteq b_{t'_h}$ *and* $Y' \subseteq b_{t'_l}$ *such that* $t'_h, t'_l \in [t_h, t_l]$*.* X *and* Y *are called the* head *and the* tail *of the sequence, respectively.*

We denote with $T_S = \langle t_{j_1}, \dots, t_{j_m} \rangle$ the *head time list* of S, i.e., the ordered list of the head's time of all the occurrences of S in the purchase history of the customer. The *support* $|T_S|$ of a sequence S is the size of its head time list. We call *length of a sequence* $|S| = |X| + |Y|$ the sum of sizes of the head and of the tail. We say that a sequence S' is a *subsequence* of S'', $S' \vec{\subseteq} S''$ if $X' \subseteq X'' \wedge Y' \subseteq Y''$.

Definition 3 (Intra-Time) *We define* $\alpha_h = t_l - t_h$ *as the* intra-time *of an occurrence of a sequence* S*, i.e., the difference between the time of the head and the time of the tail. We denote with* $A_S = \langle \alpha_1, \dots, \alpha_m \rangle$ *the ordered intra-time list of all the occurrences of* S *in* B*.*

Definition 4 (Inter-Time) *Given the head time list* T_S*, we define* $\delta_j = t_{l_i} - t_{l_j}$ *with* $t_{l_i}, t_{l_j} \in T_S$ *and* $t_{l_i} < t_{l_j}$ *as the* inter-time *of a sequence* S*, i.e., the difference between the times of the heads of two consecutive occurrences of* S*. We denote with* $\Delta_S = \langle \delta_1, \dots, \delta_m \rangle$ *the ordered inter-time list of* S*. We impose* $\delta_m = \alpha_m$ *by construction.*

To clarify the concepts defined above, let us consider the example in Table 1 which shows the purchase history of a customer. Based on the example, Figure 1 shows the occurrences of sequence $S = \{a\} \to \{b\}$. The head time list T_S consists of the times of the heads of all the occurrences of S, hence $T_S = \langle \text{01-05}, \text{01-09}, \text{01-13}, \text{01-25}, \text{02-06}, \text{02-14} \rangle$. The intra-time list A_S consists of the differences between the heads and the tails of all the occurrences of S, hence $A_S = \langle 4, 4, 16, 8, 4, 8 \rangle$. The inter-time list Δ_S consists

timestamp	basket
01-01	a, b, g, h
01-05	a, c, d
01-09	a, b, e, f, h
01-13	a, b, c, d, h
01-17	c, d, e, f, g
01-21	e, f, g

timestamp	basket
01-25	a, b, c, g, h
02-02	b, c, d
02-06	a, c, d, e, f, i
02-10	b, e, f, h
02-14	a, b, c, d, e, f, g, h
02-22	a, b, g, h, i

Table 1: Example of customer purchase history B_c.

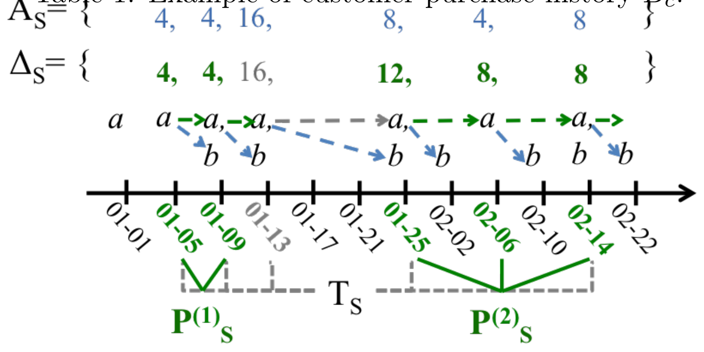


Figure 1: Head time list T_S, intra-time list A_S, inter-time list Δ_S and periods $P_S^{(1)}$, $P_S^{(2)}$ of sequence $S=\{a\}\to\{b\}$.

of all differences between the head times of two consecutive sequences, hence $\Delta_S = \langle 4, 4, 16, 12, 8, 8\rangle$. Note that: *(i)* for each $t_j \in T_S$ we have that $\alpha_j \leq \delta_j$, i.e., the intra-time of a sequence is always lower or equal than its inter-time; *(ii)* for $S = X \to X$, we have $A_S = \Delta_S$.

Definition 5 (Period) *Given a maximum inter-time δ^{max}, a minimum number of occurrences q^{min}, the head time list T_S and the inter-time list Δ_S of a sequence S, we call* period *an ordered time list $P_S^{(j)} = \langle t_h, \dots, t_l\rangle \subseteq T_S$ such that $\forall\ t_w \in P_S^{(j)}$, $\delta_w \leq \delta^{max}$, $P_S^{(j)}$ is maximal, i.e., $\delta_{h-1} > \delta^{max}$, $\delta_{l+1} > \delta^{max}$, and $|P_S^{(j)}| \geq q^{min}$. We denote with $P_S = \{P_S^{(1)}, \dots, P_S^{(m)}\}$ the set of periods of S.*

The period of a sequence S captures a temporal interval in which S occurs at least q^{min} times and the time between any two occurrences is at most δ^{max}. The support $|P_S^{(j)}|$ of a period indicates how many times S occurs in $P_S^{(j)}$. In the example of Figure 1, for $\delta^{max} = 14$ and $q^{min} = 2$ we have two periods $P_S^{(1)} = \langle 01\text{-}05, 01\text{-}09\rangle$ and $P_S^{(2)} = \langle 01\text{-}25, 02\text{-}06, 02\text{-}14\rangle$ with support 2 and 3 respectively.

Definition 6 (Recurring Sequence) *Let* $P_S = \{P_S^{(1)}, \dots, P_S^{(m)}\}$ *be a set of periods, we define* $rec(S)$=$|P_S|$ *as the* recurrence *of* S*, i.e., the number of periods* P_S *in the purchase history. Given a minimum number of periods* p^{min}*,* S *is a* recurring sequence *if* $rec(S) \geq p^{min}$*.*

In the example of Figure 1, for p^{min}=2 we have $rec(S)$=2 meaning that S is recurring. By specifying the maximum inter-time δ^{max}, the minimum number of occurrences q^{min}, and the minimum number of periods p^{min}, we can determine the set Γ_c of TARS that can be extracted from the purchase history B_c a customer c.

TARS are an evolution of both recurring patterns [16] and temporally annotated sequences [9]: the former models recurrency but do not model sequentiality and periodicity, while the latter models sequentiality and periodicity but do not model recurrency. TARS, besides co-occurrence, fills the gaps by modeling all the three aspects.

4.2 TARS Extraction Procedure

To extract the TARS from a customer's purchase history we use an extension of the well-known *FP-Growth* algorithm [10]. It builds a *FP-tree* which captures the frequency at which itemsets occur in the dataset. It has been shown in literature [2, 7, 15] that FP-Growth can be extended by attaching additional information to a FP-tree node in order to calculate the desired type of pattern.

In our approach we extend the FP-tree into a *TARS-tree*. Every node of a TARS-tree stores a sequence S, the time list T_S, its support $|T_S|$, the intra-time list A_S, the inter-time list Δ_S and the periods P_S derived from T_S with respect to δ^{max} and q^{min}.

The TARS extraction procedure is described in Algorithm 1. In the first step it extracts from the purchase history B the *base sequences* $\mathcal{S}$, i.e., the sequences of length 2 (line 1). Then, a set of parameters $\{\delta_S^{max}\}, \{q_S^{min}\}, \{p_S^{min}\}$ is estimated for each base sequence $S \in \mathcal{S}$ with respect to B (line 2). The base sequences $\mathcal{S}$ are then filtered with respect to these parameters and the base recurring sequences $\mathcal{S}^*$ are extracted, while the other base sequences are discarded to reduce the search space (line 3). Finally, the TARS-tree Ψ is built on the base recurring sequences $\mathcal{S}^*$ (line 4), and the set Γ of TARS annotated with α, p, q is extracted from the TARS-tree Ψ (line 5) according to the FP-Growth procedure.

Algorithm 1: $extractTars(B)$

```
1 $\mathcal{S} \leftarrow extractBaseSequences(B)$;
2 $\{\delta_S^{max}\}, \{q_S^{min}\}, \{p_S^{min}\} \leftarrow parametersEstimation(B, \mathcal{S})$;
3 $\mathcal{S}^* \leftarrow sequenceFiltering(B, \mathcal{S}, \{\delta_S^{max}\}, \{q_S^{min}\}, \{p_S^{min}\})$;
4 $\Psi \leftarrow buildTars\text{-}Tree(B, \mathcal{S}^*, \{\delta_S^{max}\}, \{q_S^{min}\}, \{p_S^{min}\})$;
5 $\Gamma \leftarrow extractTarsFromTree(\Psi)$;
6 return $\Gamma$;
```

4.2.1 Data-Driven Parameters Estimation

In order to make the parameters δ^{max}, q^{min}, p^{min} adaptive not only to the individual customer [14], but also to the various sequences in B_c, we apply two pre-processing steps on the base sequences (lines 1–2 Algorithm 1). The first pre-processing step is the data-driven estimation of the sets of parameters $\{\delta_S^{max}\}$, $\{q_S^{min}\}$, $\{p_S^{min}\}$ described in Algorithm 2.

Let $\mathcal{S}$ be the set of base sequences and $\hat{\delta_S}$ be the median of inter-times in Δ_S (Algorithm 2, line 2). Given a base sequence S, we estimate parameter δ^{max} by the following two steps: *(i)* we group the base sequences with similar inter-times $\hat{\delta_S}$ (line 3) obtaining a set of clusters $\mathcal{C}_{\delta^{max}}=\{C_1, \dots, C_v\}$; *(ii)* if $S \in C_h$, $C_h \in \mathcal{C}_{\delta^{max}}$, we set δ_S^{max} as the median of the $\hat{\delta_S}$ values in cluster C_h (lines 4–5).

Then, we calculate the periods TC_S compliant only with the temporal constraint δ_S^{max} (lines 6–8) and we estimate $\{q_S^{min}\}$ as follows: *(i)* we group the base sequences with similar median number of occurrences per period $\hat{q_S}$, producing a set of clusters $\mathcal{C}_{q^{min}} = \{C_1, \dots, C_g\}$ (line 9); and *(ii)* if $S \in C_h$, $C_h \in \mathcal{C}_{q^{min}}$ we set q_S^{min} as the median of the $\hat{q_S}$ in C_h (lines 10–11).

Similarly, we estimate $\{p_S^{min}\}$ as follows: *(i)* we compute the sum of the number of occurrences of a base sequence in the periods w_S and we calculate the expected number of occurrences per period e_S as $w_S/|P_S|$ (lines 12–14); *(ii)* we group the base sequences with similar e_S producing a set of clusters $\mathcal{C}_{p^{min}} = \{C_1, \dots, C_d\}$ (line 15); and *(iii)* if $S \in C_h$, $C_h \in \mathcal{C}_{p^{min}}$, we set p_S^{min} as the median of the number of periods of the base sequences in C_h (lines 16–17).

We group the base sequences by dividing the values into equal-sized bins [21], whose number is estimated as the maximum between the estimated number of bins suggested by the Sturges [26] and the Freedman-Diaconis methods [8].

Algorithm 2: *parametersEstimation*$(\mathcal{S}, B)$

```
1  $D_{\delta^{max}} \leftarrow \emptyset$; $D_{q^{min}} \leftarrow \emptyset$; $D_{p^{min}} \leftarrow \emptyset$;
2  foreach $S \in \mathcal{S}$ do $D_{\delta^{max}} \leftarrow D_{\delta^{max}} \cup \{\hat{\delta_S} = median(\Delta_S)\}$;
3  $\mathcal{C}_{\delta^{max}} \leftarrow groupSimilar(D_{\delta^{max}})$;
4  for $C_h \in \mathcal{C}_{\delta^{max}}$ do
5      foreach $S$ $assignedTo(C_h)$ do $\delta_S^{max} \leftarrow median(C_h)$;
6  for $S \in \mathcal{S}$ do
7      $TC_S \leftarrow getTemporalyCompliantPeriods(S, B, \{\delta_S^{max}\})$;
8      $D_{q^{min}} \leftarrow D_{q^{min}} \cup \{median(\{\hat{q_S} = |TC_S^{(j)}|$ s.t. $TC_S^{(j)} \in TC_S\})\}$;
9  $\mathcal{C}_{q^{min}} \leftarrow groupSimilar(D_{q^{min}})$;
10 for $C_h \in \mathcal{C}_{q^{min}}$ do
11     foreach $S$ $assignedTo(C_h)$ do $q_S^{min} \leftarrow median(C_h)$;
12 for $S \in \mathcal{S}$ do
13     $P_S \leftarrow getPeriods(S, B, \{\delta_S^{max}\}, \{q_S^{min}\})$;
14     $w_S \leftarrow \sum_{P_S^{(j)} \in P_S} |P_S^{(j)}|$; $e_S \leftarrow w_S/|P_S|$; $D_{p^{min}} \leftarrow D_{p^{min}} \cup \{e_S\}$
15 $\mathcal{C}_{p^{min}} \leftarrow groupSimilar(D_{p^{min}})$;
16 for $C_h \in \mathcal{C}_{p^{min}}$ do
17     for $S$ $assignedTo(C_h)$ do
18         $p_S^{min} \leftarrow median(\{rec(P_{S'}) = |P_{S'}|$ s.t. $S' assignedTo(C_h)\})$;
19 return $\{\delta_S^{max}\}, \{q_S^{min}\}, \{p_S^{min}\}$;
```

4.2.2 Sequence Filtering

The second pre-processing step consists in the selection of the *base recurring sequences*, i.e., the base sequences satisfying the sets of parameters $\{\delta_S^{max}\}$, $\{q_S^{min}\}$, $\{p_S^{min}\}$. We apply this filtering in order to reduce the search space so that the building of the TARS-tree and the TARS extraction (lines 4–5 Algorithm 1) are employed only on the super-sequences of the base recurring sequences. In other words, if S_1 is not a base recurring sequence and $S_1 \vec{\subseteq} S_2$, then we assume as heuristic that S_2 is not recurring too, and we eliminate it through the sequence filtering process. We adopt the sequence filtering heuristic for reducing the search space because the *antimonotonic property* [1] does not apply to TARS. Consider $S_1 = \{c\} \rightarrow \{c\}$ and $S_2 = \{c, d\} \rightarrow \{c\}$ in the example of Table 1. We have that $S_1 \vec{\subseteq} S_2$. Given δ^{max}=14, q^{min}=2 and p^{min}=2, we have $rec(S_1)$=1 and $rec(S_2)$=2. Hence, S_2 is recurrent while S_1 is not, and the anti-monotonic property is not satisfied.

However, it is clear from this example that a TARS with S_1 could be

Algorithm 3: $getActiveTARS(B, t_{n+1}, \Gamma)$

```
1  $\hat{\Gamma} \leftarrow \emptyset$; $Q \leftarrow \emptyset$; $L \leftarrow \emptyset$; $\Upsilon \leftarrow \Gamma$;
2  for $b_{t_j}, b_{t_{j-1}} \in$ sort-desc$(B)$ do
3      $\alpha_{j-1} \leftarrow t_j - t_{j-1}$;
4      for $X \subseteq b_{t_{j-1}}$ do
5          for $Y \subseteq b_{t_j}$ do
6              if $\exists\, \gamma \in \Upsilon \mid \gamma = (S, \alpha, p, q) \wedge \alpha_1 \leq \alpha_{j-1} \leq \alpha_2 \wedge$
                   $S = \langle X, Y \rangle = X \rightarrow Y$ then
7                  if $\gamma \in \hat{\Gamma}$ then
8                      $Q_\gamma \leftarrow Q_\gamma + 1$; $L_\gamma \leftarrow t^{j-1}$;
9                      if $Q_\gamma > q$ then $\hat{\Gamma} \leftarrow \hat{\Gamma}/\{\gamma\}$; $\Upsilon \leftarrow \Upsilon/\{\gamma\}$;
10                     if $L_\gamma - t_{j-1} > q \cdot (\alpha_1$-$\alpha_2)$ then $\Upsilon \leftarrow \Upsilon/\{\gamma\}$;
11                 else
12                     $\hat{\Gamma} \leftarrow \hat{\Gamma} \cup \{\gamma\}$; $Q_\gamma \leftarrow 1$; $L_\gamma \leftarrow t_{j-1}$;
13                 if $\Upsilon = \emptyset$ then return $\hat{\Gamma}, Q$;
14 return $\hat{\Gamma}, Q$;
```

useful for the prediction because, despite $rec(S_1) = 1$ in total it occurs six times $|P^{(1)}_{S_1}| = 6$. In real-world, $\{c\}$ could be a fresh product (like milk or salad) that is repeatedly and frequently purchased. Hence, an imposed parameter setting could be not appropriate because *(i)* it could remove too many TARS which are in fact useful for the prediction; *(ii)* it could consider too many valid base sequences and not prune enough the search space.

For these reasons we developed the two pre-processing steps heuristic for parameters estimation described in this section.

5 TARS Based Predictor

On top of the set Γ_c of TARS extracted from the purchase history B_c of customer c we build the *TARS Based Predictor* (*TBP*), an approach for next basket prediction that is markedly *personalized* and *user-centric* [12, 22], in the sense that just the model build on the individual purchase history B_c of customer c, i.e., her TARS Γ_c, is used to make the predictions about that customer c.

TBP exploits TARS to simultaneously embed complex item interactions such as the co-occurrence (which item is bought with which), sequential relationship (which items are bought after which), periodicity (which item is bought when) and typical times of re-purchase (after when re-purchases happen). These factors enable TBP to observe the recent purchase history of a customer and understand which are the *active* patterns, i.e., the patterns that the customer is currently following in her purchasing. In turn, by realizing which are the active patterns TBP can provide the set of items that she will need at the time of the next purchase. It is worth noting that TBP is parameter-free: all the parameters of the TARS model Γ_c are automatically estimated for each customer on her personal data B_c, avoiding the usual case where the same parameter setting is used indiscriminately for all the customers [14].

Given the purchasing history B_c of customer c, the time t_{n+1} of c's next transaction, and c's TARS set Γ_c, the TBP approach works in two steps. First, it selects the set $\hat{\Gamma}_c$ of *active* TARS. Second, it computes a score Ω_{c_i} for every item i belonging to a TARS in $\hat{\Gamma}_c$, ranks the items according to their score Ω_{c_i}, and selects the top k items as the basket prediction for customer c.

Algorithm 3 shows the procedure of the TBP to select the *active* TARS of a customer $\hat{\Gamma}$. First, it sorts the purchase history B ordering it chronologically from the most recent basket to the oldest one, then it loops on pairs of consecutive baskets (line 2) searching for a set Υ of *potentially active* TARS (lines 4–7). When it finds a potentially active TARS γ, it considers two cases. If the sequence S of γ is encountered for the first time, the algorithm adds γ to the set $\hat{\Gamma}$ of active TARS and initializes two variables: the number of times γ has been encountered Q_γ and its last starting time L_γ (line 13). In the second case, the algorithm increments Q_γ and updates L_γ (line 9). If $Q_\gamma > q$ the algorithm removes γ from the set of active TARS and from the set of potentially active TARS (line 9). If too much time has passed between the last beginning of TARS γ and its next occurrence (line 11), the algorithm does not look for that TARS γ anymore and removes it from Υ. Algorithm 3 stops either when the set of potentially active TARS is empty (line 14), or when the entire purchase history B has been scanned (line 15). Finally, it returns the set $\hat{\Gamma}$ of active TARS and the number of times Q the sequences of the active TARS have occurred in the last period.

Algorithm 4 shows the procedure of TBP to compute the items' scores. First, it sets to zero the score of each item Ω_i (line 1) Then, for every active TARS γ containing item $i \in Y$, it increases Ω_i with the difference between the typical number of occurrences q of γ and Q_γ indicating the number of times that the sequence of γ occurred in the recent history (lines 2–

Algorithm 4: *calculateItemScore*$(B, \hat{\Gamma}, Q)$

```
1 $\Omega \leftarrow \emptyset$; foreach $i \in I$ do $\Omega_i \leftarrow 0$;
2 for $\gamma = (S = \langle X, Y \rangle, \alpha, p, q) \in \hat{\Gamma}$ do
3     foreach $i \in Y$ do $\Omega_i \leftarrow \Omega_i + (q - Q_\gamma)$;
4 for $i \in \{i \mid \exists\, \gamma = (S = \langle X, Y \rangle, \alpha, p, q) \in \hat{\Gamma}, i \in Y\}$ do
5     $\Omega_i \leftarrow \Omega_i + sup(i)$
6 return $\Omega$;
```

3). Finally, Ω_i is augmented with the support of item i for the items in the tail of the active TARS (lines 4–5).

After this procedure TBP ranks the items' scores Ω_c in descending order and returns the top-k items as the predicted basket.

6 Experiments on Retail Data

In this section, we report the experiments performed on three real-world datasets in order to show the properties of the TARS and the effectiveness of TBP in next basket prediction.

State-of-the-art methods [24, 31, 32, 4] fix the size of the predicted basket to k=5 or k=10. However, we think that the size k of the predicted basket should adapt to the customer's personal behavior. Indeed, if a customer typically purchases baskets with a few items it is useless to predict a basket with a large number of items. On the other hand, if a customer typically purchases baskets with a large number of items, the prediction of a small basket will not cover most of the items purchased. In this paper we report the evaluation of the predictions made using both a fixed length $k \in [2, 20]$ for all the customers and using a customer-specific size $k = k^*_c$, where k^*_c indicates the average basket length of customer c.

According to the literature, we adopt a *leave-one-out* strategy for model validation [32, 31, 24, 4]: for each customer c we use the baskets in the purchase history $B_c = \{b_{t_1}, \dots, b_{t_n}\}$ for extracting the TARS, and the basket $b_{t_{n+1}}$ as test to estimate the performance.

For every customer, we evaluate the agreement of the predicted basket b^* and the real basket b by using the following metrics:

- *F1-score*, the harmonic mean of precision and recall [28]:

$$F1\text{-}score(b, b^*) = \frac{2 \cdot Precision(b, b^*) \cdot Recall(b, b^*)}{Precision(b, b^*) + Recall(b, b^*)}$$

Dataset	***cust.***	***# baskets***	***# items***	***avg basket per cust.***	***avg basket length***
Coop-A	10,000	7,407,056	4,594	432.4±353.4	9.4±5.8
Coop-C	10,000	7,407,056	407	432.4±353.4	8.6±4.9
Ta-Feng	2,319	24,304	5,117	10.4±7.5	1.8±1.1

Table 2: Statistics of the datasets used in the experiments.

$$Precision(b, b^*) = |b \cap b^*|/|b^*| \quad Recall(b, b^*) = |b \cap b^*|/|b|$$

- *Hit-Ratio*, the ratio of customers who received at least one correct prediction (a *hit*) [13]:

$$\textit{Hit-Ratio}(b, b^*) = 1 \text{ if } b \cap b^* \neq \emptyset, 0 \text{ otherwise.}$$

- *normalized F1-score*: the F1-score calculated only for the customers having at least one hit.

Furthermore, for each customer we compute both *learning time* and *prediction time*. The learning time is the amount of time required to extract the model from the data. The prediction time is the amount of time the predictor needs to predict the next basket of a customer. We perform the experiments on Ubuntu 16.04.1 LTS 64 bit, 32 GB RAM, 3.30GHz Intel Core i7.

6.1 Datasets

For our experiments we use three real-world transactional datasets: *Coop-A*, *Coop-C* (both extracted from the *Coop* repository) and *Ta-Feng*. Table 2 shows the details of the datasets.

The *Coop* repository is provided by *Unicoop Tirreno*[1], a big retail supermarket chain in Italy. It stores 7,407,056 transactions made by 10,000 customers in 23 different shops in the province of Leghorn, over the years 2007-2014. The set of Coop items includes food, household, wellness, and multimedia items. There are 7,690 different articles classified into 520 market categories. From the repository we extract two datasets: *Coop-A* and *Coop-C*. The two datasets differ in the items categorization. In ***Coop-A*** (articles) the items of a basket are labeled with a fine-grained categorization which distinguishes, for example, between blood orange and navel orange. In ***Coop-C*** (categories) the items are mapped to a more general category: in

[1]https://www.unicooptirreno.it/

the example above blood orange and navel orange are considered the same generic item (orange). All the customers in *Coop-A* and *Coop-C* have at least one purchase per month.

Ta-Feng[2] dataset covers food, office supplies and furniture, with a total of 23,812 items. It contains 817,741 transactions made by 32,266 customers over four months. We remove the customers with less than 10 baskets and consider the remaining 7% customers.

Since we do experiments on real retail datasets we adopt the *day* as time unit, i.e., parameters and annotations are expressed in days.

6.2 Interpretability of TARS

The interpretability of TARS is one of the main characteristics of our approach. Table 4 shows some examples of TARS extracted from *Coop-C*. In the table we report the median of α, p and q across all the customers having the presented TARS. We observe that TARS with a recurring base sequence are the most supported among the customers. For example $\{milk\}\to\{milk\}$ and $\{banana\}\to\{banana\}$ are supported by more than 90% of the customers in *Coop-C*. The two TARS have similar q (6.58 and 7.20 respectively) indicating that they have similar recurrence degrees, i.e., they occurs a similar number of times in the respective periods. In contrast $\{banana\}\to\{banana\}$ has a higher maximum intra-time (α_2=35) and a lower average number of recurrences (p=14.63). This indicates that: *(i)* the time for a banana re-purchase is higher than the time of a milk re-purchase; *(ii)* the support to have a distinct period is higher for $\{banana\}$ than $\{milk\}$. We notice for more than 25% of the customers the contemporary purchase $\{bread, tomato\}$ can indicate a future basket with $\{bovine\}$ or with $\{banana, potato\}$ and that these TARS have very different annotations α, p, q. Finally, we highlight that, even if the most common TARS among the customers are those with base sequences, the TARS in Γ_c with sequence length greater than two are on average more than the 95% for each customer.

To better understand the TARS, in Table 3 we shows some TARS made of base recurring sequences with different peculiarities. A base recurring sequence capture the typical repurchasing of the same item within a certain period for a certain number of times. *Apples* and *bananas* are fruit items available throughout they year. The associated base TARS $\{banana\}\to\{banana\}$ and $\{apple\}\to\{apple\}$ have indeed a similar number of periods p and number of typical occurrences in each period q. On the other hand, *oranges* are a sea-

[2]http://www.bigdatalab.ac.cn/benchmark/bm/dd?data=Ta-Feng

- Supported by more than 90% customers

$$\{\text{milk}\} \xrightarrow[18.87,6.58]{(1,17)} \{\text{milk}\} \quad \{\text{banana}\} \xrightarrow[14.63,7.20]{(2,35)} \{\text{banana}\}$$

- Supported by more than 80% customers

$$\{\text{tomato}\} \xrightarrow[13.87,6.58]{(1,17)} \{\text{milk}\} \quad \{\text{tomato}\} \xrightarrow[15.27,5.11]{(1,12)} \{\text{bovine}\}$$

- Supported by more than 25% customers

$$\left\{\begin{matrix}\text{bread,}\\ \text{potato}\end{matrix}\right\} \xrightarrow[11.40,8.15]{[2,15]} \{\text{bovine}\} \left\{\begin{matrix}\text{bread,}\\ \text{potato}\end{matrix}\right\} \xrightarrow[7.25,4.30]{[3,27]} \left\{\begin{matrix}\text{banana,}\\ \text{potato}\end{matrix}\right\}$$

Table 3: Examples of TARS extracted from *Coop-C*.

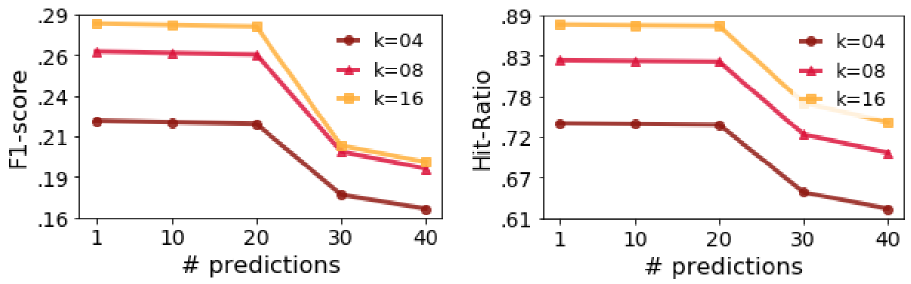
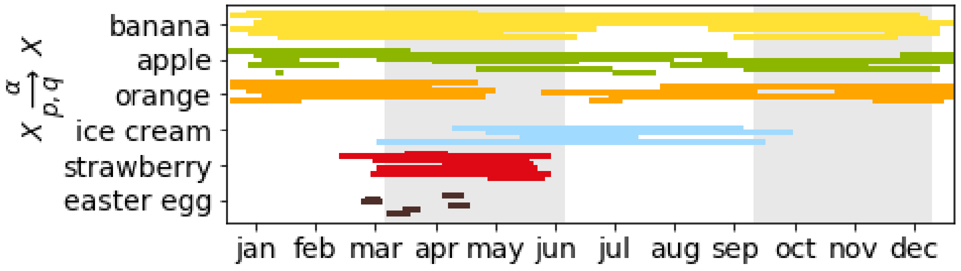


Figure 2: Evaluation of TARS temporal validity on *Coop-C*.

sonal fruit item, generally available between November and February. The associated base TARS $\{orange\}\rightarrow\{orange\}$ has a recurrence p significantly lower than the recurrence of banana and apple TARS, while the occurrence inside a period is similar. We observe that ice creams are similar to oranges: the associated TARS $\{ice\ cream\}\rightarrow\{ice\ cream\}$ has a lower p and a higher maximum intra time α_2. Finally, *Strawberries* and *Easter eggs* are items available for just a short period of the year. As reflection in the associated TARS we have lower values of both p and q than the other TARS. In particular, among the items considered strawberries' TARS have the lowest α_2 indicating short periods, while Easter eggs have the highest α_1 indicating long intra-times.

6.3 Properties of TBP

In this section we present some peculiar properties of TBP: we show the temporal validity and reliability of the TARS extracted, and the performance improvements yield by the parameters evaluation.



X	$\rightarrow$	X	α_1	α_2	p	q
{banana}	$\rightarrow$	{banana}	2	35	14.63	7.20
{apple}	$\rightarrow$	{apple}	2	35	15.90	6.14
{orange}	$\rightarrow$	{orange}	2	33	8.13	6.56
{ice cream}	$\rightarrow$	{ice cream}	2	40	5.90	6.38
{strawberry}	$\rightarrow$	{strawberry}	2	32	3.55	4.69
{easter egg}	$\rightarrow$	{easter egg}	4	20	2.42	3.29

Table 4: TARS with different recurring base sequences from *Coop-C*, and their periods shown along 7 years of observations (each year represented as a single line).

6.3.1 TARS Temporal Validity

In real-world applications it is unpractical (or even unnecessary) to rebuild a predictive model from scratch every time a new basket appears in a customer's purchase history. This leads to the following question: for how long are TBP predictions reliable? We address this question by extracting TARS on the 70% of the purchase history of every customer and performing the prediction on the subsequent baskets. As shown in Figure 2, regardless the predicted basket size k, F1-score and Hit-Ratio remain stable up to 20 predictions, which suggests a large temporal validity of TBP since the model construction.

6.3.2 TARS Extraction Reliability

How many baskets does TBP need to perform reliable predictions? For each customer we start from her second week of purchases and extract TARS incrementally by extending the training set one week at a time. We then predict the next basket of the customer and evaluate the performance of TBP in this scenario. Figure 3 shows the median value and the "variance" (by means of the 10th, 25th, 75th and 90th percentiles) of the F1-score (top-

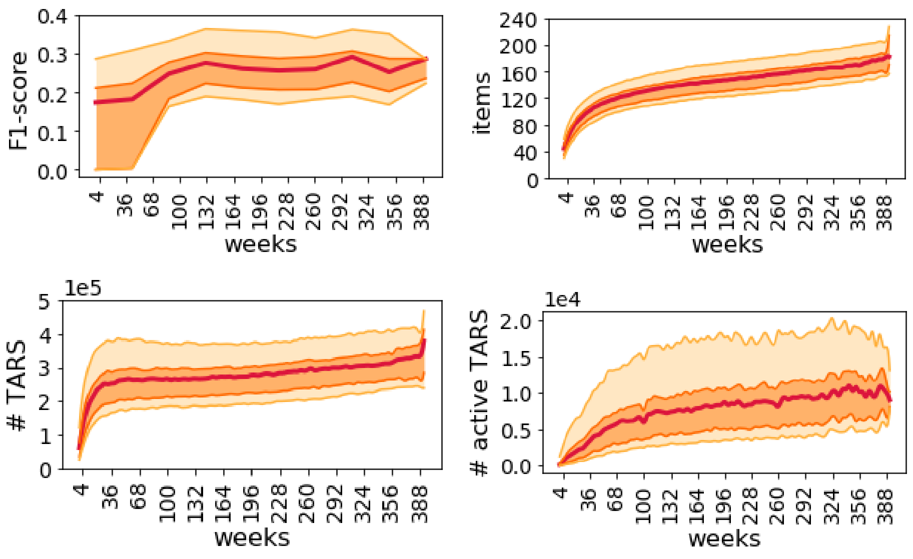


Figure 3: Evaluation of the TARS reliability by temporally augmenting the purchase history on *Coop-C*.

left), the total number of different items purchased by the customer (top-right), the number of TARS extracted (bottom-left), the number of active TARS during the prediction (bottom-right) as the number of weeks used in the learning phase increases. On one hand the average F1-score does not change significantly as the number of weeks increases, while its "variance" reduces as more weeks are used in the learning phase. On the other hand, the other three measures stabilize after an initial setup phase.

6.3.3 Parameter-Free vs. Parameter-Fixed Approach

TARS can be extracted by fixing the same parameters for all the customers and items, as usually done by state-of-the-art methods [32, 31, 24, 4], or by automatically estimating the parameters with a data driven procedure. Here we discuss the impact of fixing the parameters on the predictive performance by comparing the results of parameter-free TBP and a parameter-fixed version of TBP where we set $\delta^{max} = 14$ (e.g., two weeks), $q^{min} = 3$ and $p^{min} = 2$. Figure 4 shows the distributions of the number of TARS per customer for the parameter-free (left) and parameter-fixed (right) scenarios. We observe two different distributions: a skewed peaked distribution for the parameter-free scenario and a heavy tail distribution for the parameter-fixed

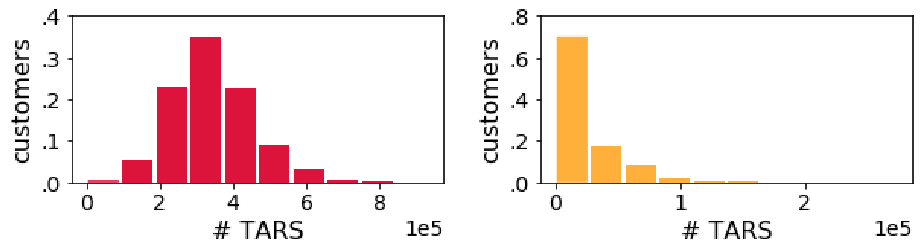


Figure 4: Number of TARS per customer distribution on *Coop-C*: parameter-free (left), parameter-fixed (right).

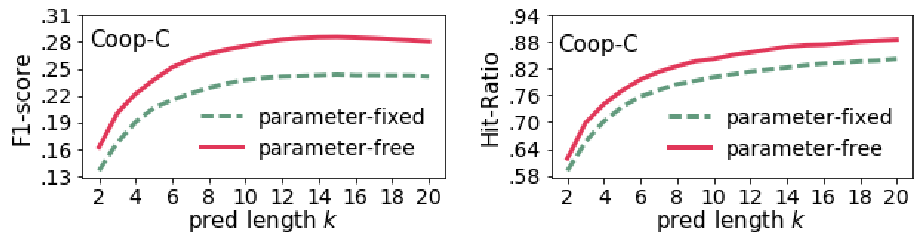


Figure 5: Next basket prediction performance on *Coop-C*: parameter-free (left), parameter-fixed (right).

scenario. This suggests that fixing the parameters has a strong impact on the extraction of TARS, leading to a lower average number of TARS per customer than the parameter-free scenario (Figure 4).

Figure 5 compares the predictive performances of the parameter-free and the parameter-fixed scenarios. For both F1-score and Hit-Ratio, TBP produces better predictions in the parameter-free scenario. In particular, when using the average basket size of a customer k_c^* as the size of the predicted basket, the parameter-free approach has F1-score=0.25 while the parameter-fixed approach has F1-score=0.21. Our results suggest that the adoption of a parameter-free strategy during the extraction of TARS enforces customer behavior heterogeneity and increases prediction accuracy.

6.4 Comparison with Baseline Methods

We compare TBP with several baseline methods on the three datasets described above. To this purpose, we implement[3] the following user-centric

[3]We provide the Python 2.7.11 source code of TBP and the baseline methods along with an anonymized sample of the *Coop* dataset at this link https://goo.gl/JVsJcP. The

state-of-the-art methods:

LST [4]: the next basket predicted is just the *last* basket purchased by the customer, i.e., $b_{t_{n+1}} = b_{t_n}$;

TOP [4]: predicts the top-k *most frequent* items with respect to their appearance in the customer's purchase history B_c;

MC [4]: makes the prediction based on the last purchase b_{t_n} and on a *Markov chain* calculated on the customer's purchase history B_c;

CLF [4]: for each item i purchased by the customer, this method builds a *classifier* on temporal features extracted from the customer's purchase history considering two classes: "item i purchased yes/no". The classifier then predicts the next basket by using the temporal features extracted from the customer's purchase history.

We also implement four state-of-the-art methods that are not user-centric, i.e, they require purchase data of all customers to build the predictive model for a single customer:

NMF (*Non-negative Matrix Factorization*) [19]: is a collaborative filtering method which applies a non-negative matrix factorization over the customers-items matrix. The matrix is constructed from the purchase history of all customers;

FMC (*Factorizing personalized Markov Chain*)[24]: combines Markov chains and matrix factorization to predict the next basket based on the purchase history of all the customers $\mathcal{B}$;

HRM (*Hierarchical Representation Model*) [31]: employs a two-layer structure to construct a hybrid representation over customers and items purchase history $\mathcal{B}$ from last transactions;

DRM (*Dynamic Recurrent basket Model*) [32]: it is based on recurrent neural network and captures both sequential features from all the baskets of a customer, and global sequential features from all the baskets of all the customers $\mathcal{B}$.

We compare TBP with the above defined baselines on *Coop-A*, *Coop-B* and *Ta-Feng*. For NMF, FMC, HRM and DRM we report the results obtained with the default parameters setting [32, 31] and a dimensionality d=100 for *Ta-Feng*, *Coop-A* and *Coop-C*. Even though the methods [11, 18, 30] employ patterns for producing recommendations we do not compare against them because they are systems mainly designed for web-based data services and because they are also consider the items' ratings and not only the implicit feedbacks provided by the presence of the items in a basket.

Figure 6 compares the average F1-score (left) and the average Hit-Ratio

code of DRM was kindly provided by the authors of [32].

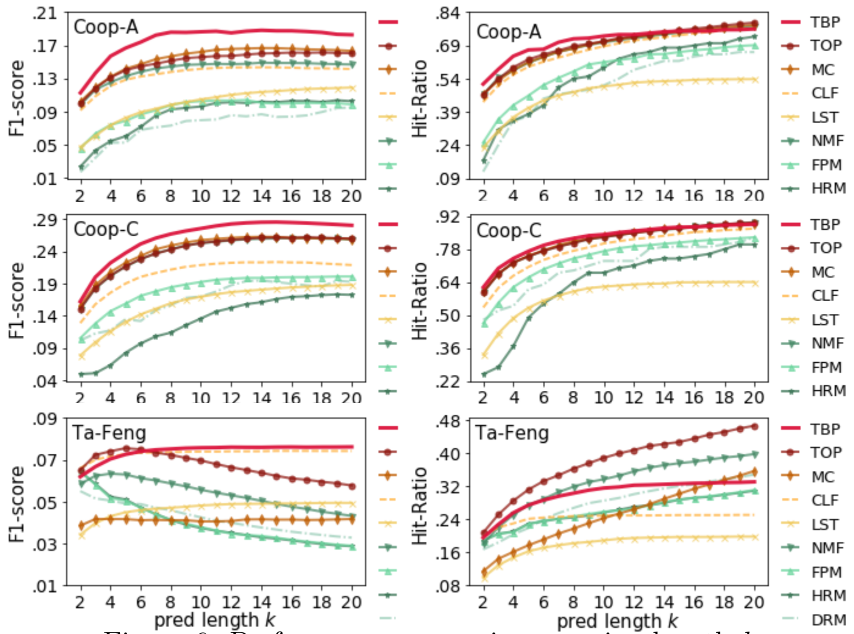


Figure 6: Performance comparison varying length k.

(right) produced by all the methods, varying $k \in [2, 20]$. We observe that TBP outperforms the competitors on *Coop-A* and *Coop-B*, having the highest F1-score and a comparable Hit-Ratio. On *Ta-Feng* TBP has the highest F1-score at the third highest Hit-Ratio. The performance of TBP significantly improves, both in terms of F1-score and Hit-Ratio, when we use $k = k_c^*$, as shown in Table 5.

The decrease of the Hit-Ratio of TBP in *Ta-Feng* can be due to its high data sparsity. As we observe in Table 2, *Ta-Feng* has a much lower average number of baskets per customer, a much lower average basket length, and a shorter observation period than *Coop-A* and *Coop-C*. For this reason the TARS extracted from *Ta-Feng* have lower quality than the TARS extracted on the other datasets.

In Table 6 we report the duration of the learning process, i.e., the execution time needed to build every method on the four datasets. For the user-centric methods (TBP, MC, CLF) we report the average execution time per customer, while we report the total execution time for not user-centric methods (NMF, FPM, FRM, DRM). We do not report the prediction time because it is negligible for all the approaches (i.e., less than 0.01 seconds). We observe that TBP needs more time than existing user-centric methods (5 minutes per customer on average) but it is much faster than the not user-

	$k = k_c^*$	TBP	TOP	MC	CLF	LST	NMF	FPM	HRM	DRM
F1-score	*Coop-A*	**.17**	***.14***	***.14***	.13	.09	***.14***	.08	.06	.05
	Coop-C	**.24**	.22	***.23***	.19	.14	.22	.16	.08	.12
	Ta-Feng	**.07**	**.07**	.04	**.07**	.04	***.06***	***.06***	***.06***	.05
Hit-Ratio	*Coop-A*	**.62**	.58	.58	.56	.40	***.59***	.44	.35	.33
	Coop-C	**.72**	***.71***	.70	.65	.50	***.71***	.61	.38	.55
	Ta-Feng	.20	**.24**	.14	***.21***	.15	***.21***	***.21***	***.21***	.19

Table 5: Performance using personalized length $k = k_c^*$. In **bold**, and ***bold-italic*** the 1st and 2nd best performer.

Dataset	TBP	MC	CLF	NMF	FPM	HRM*	DRM*
Coop-A	351.86s	0.04s	2.38s	244.28s	0.21h	0.84h	47.53h
Coop-C	6.62s	0.01s	1.08s	69.98s	0.11h	0.72h	34.06h
Ta-Feng	0.01s	0.00s	0.00s	803.89s	0.41h	0.34h	4.24h

Table 6: Building time comparison. Note that: *(i)* the time is reported in seconds (s) or in hours (h), *(ii)* for individual methods we report the average building time, for collective methods the total, *(iii)* the building time for TOP and LST is always lower than 0.01 seconds and is not reported.

centric approaches. We believe that such a learning time is acceptable for two reasons: *(i)* in a real scenario the TARS can be re-computed once every month and still produce reliable predictions; *(ii)* the computation can be parallelized and personalized with respect to the customer's behavior, thus the TARS of all the customers can be extracted at the same time by different devices.

It is worth noting that the value of the average F1-score can be biased by two extreme scenarios: *(i)* the F1-score can be low because of a low Hit-Ratio, i.e., for most of the customers no item is predicted even though for some customers we predict most of the items; *(ii)* the F1-score can be high because for most of the customers just one item is predicted. In Figure 7 we show the results of the experiments using the normalized F1-score instead of the simple F1-score. We observe that the positive gap between TBP and the competitors increases: for the customers for which TBP correctly predicts at least one future basket, the baskets predicted by TBP are more accurate and cover a larger number of items than the baskets predicted by the other methods.

We also investigate at what extent the performances can be affected by

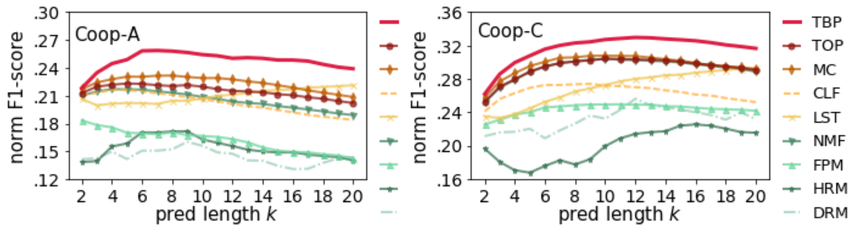


Figure 7: Normalized F1-score varying length k.

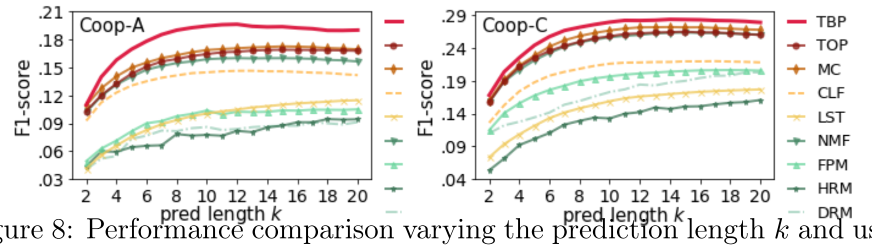


Figure 8: Performance comparison varying the prediction length k and using a model built on a subset of B_c having random size between 70% and 90% of $|B_c|$.

the *leave-one-out* evaluation strategy: the last basket of a customer could depart from her typical behavior affecting the extraction of the TARS. To cope with this issue we also perform the learning process (i.e., extract TARS) by selecting a random subset $B'_c = \{b_{t_1}, \ldots, b_{t_{n'}}\}$ of the customers' purchase history $B_c = \{b_{t_1}, \ldots, b_{t_n}\}$, with $t_{n'} < t_n$. We randomly vary the size of the subset $|B'_c|$ among 70% and 90% of $|B_c|$, and we apply TBP on the subsequent basket $b_{t_{n'+1}}$. Figure 8 presents the results of this experiment for *Coop-A* and *Coop-C* and confirms the trends observed on the previous experiments indicating that the *leave-one-out* evaluation strategy does not affect significantly the performance of the methods.

7 Conclusions

In this work we propose a data-driven and user-centric approach for next basket prediction. Our contribution is twofold. First we define Temporal Annotated Recurring Sequences (TARS). Then we then use TARS to build a predictor for forecasting customers' next baskets. Being parameter-free, TBP leverages the specificity of individual customer's behavior to adjust the way TARS are extracted, thus producing more personalized patterns. We perform experiments on real-world datasets, show that TBP outperforms state-of-the-art methods and, in contrast with them, it provides interpretable patterns that can be used to gather insights on customers' shopping behaviors. Our results show that at least 36 weeks of a customer's purchase behavior are needed to effectively predict her next baskets. In this scenario, TBP can effectively predict the subsequent twenty future baskets with remarkable accuracy.

A future research line consists in providing to the customers of a living laboratory [29] an app running TBP and observe how and if their purchase behaviors are influenced by the recommendations. Moreover, since our method is fully user-centric, it cannot make reliable predictions for new customers or for customers having a short purchase history. Thus, we plan to build a version of TBP which incorporates a collaborative filtering approach, such that it will be able to forecast baskets for newcomers and for customer with a short purchase history. Finally, we would like to exploit TARS also to segment the customers, and to investigate how TARS and TBP can be applied on different analytical domains such as for mobility and for health data analysis.

This work is partially supported by the GS501100001809 European Communitys H2020 Program under the funding scheme "INFRAIA-1-2014-2015: Research Infrastructures" grant agreement, http://www.sobigdata.eu, GS501100001809, 654024 *"SoBigData: Social Mining & Big Data Ecosystem"*.

References

[1] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *Sigmod Record*, number 2, pages 207–216. ACM, 1993.

[2] K. Amphawan, A. Surarerks, and P. Lenca. Mining periodic-frequent itemsets with approximate periodicity using interval transaction-ids list tree. In *SIGKDDw*, pages 245–248. IEEE, 2010.

[3] C. Chand, A. Thakkar, and A. Ganatra. Sequential pattern mining: Survey and current research challenges. *IJSCE*, 2(1):185–193, 2012.

[4] C. Cumby, A. Fano, R. Ghani, and M. Krema. Predicting customer shopping lists from point-of-sale purchase data. In *SIGKDD*, pages 402–409. ACM, 2004.

[5] A. Dagher. Shopping centers in the brain. *Neuron*, 53(1):7–8, 2007.

[6] Y.-A. de Montjoye, E. Shmueli, S. S. Wang, and A. S. Pentland. openpds: Protecting the privacy of metadata through safeanswers. *PloS one*, 9(7):e98790, 2014.

[7] P. Fournier-Viger, J. C.-W. Lin, Q.-H. Duong, and T.-L. Dam. Phm: mining periodic high-utility itemsets. In *ICDM*, pages 64–79. Springer, 2016.

[8] D. Freedman and P. Diaconis. On the histogram as a density estimator: L 2 theory. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 57(4):453–476, 1981.

[9] F. Giannotti, M. Nanni, and D. Pedreschi. Efficient mining of temporally annotated sequences. In *SDM*, pages 348–359. SIAM, 2006.

[10] J. Han, J. Pei, and Y. Yin. Mining frequent patterns without candidate generation. In *Sigmod Record*, volume 29, pages 1–12. ACM, 2000.

[11] C.-N. Hsu, H.-H. Chung, and H.-S. Huang. Mining skewed and sparse transaction data for personalized shopping recommendation. *ML*, 57(1-2):35–59, 2004.

[12] C. Kalapesi. Unlocking the value of personal data: From collection to usage. In *World Economic Forum technical report*, 2013.

[13] G. Karypis. Evaluation of item-based top-n recommendation algorithms. In *CIKM*, pages 247–254. ACM, 2001.

[14] E. Keogh, S. Lonardi, and C. A. Ratanamahatana. Towards parameter-free data mining. In *SIGKDD*, pages 206–215. ACM, 2004.

[15] R. U. Kiran and M. Kitsuregawa. Finding periodic patterns in big data. In *BDA*, pages 121–133. Springer, 2015.

[16] R. U. Kiran, H. Shang, M. Toyoda, and M. Kitsuregawa. Discovering recurring patterns in time series. In *EDBT*, pages 97–108, 2015.

[17] B. Knutson, S. Rick, G. E. Wimmer, D. Prelec, and G. Loewenstein. Neural predictors of purchases. *Neuron*, 53(1):147–156, 2007.

[18] E. Lazcorreta, F. Botella, and A. Fernández-Caballero. Towards personalized recommendation by two-step modified apriori data mining algorithm. *Expert Systems with Applications*, 35(3):1422–1429, 2008.

[19] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2001.

[20] B. Mittal and W. M. Lassar. The role of personalization in service encounters. *Journal of Retailing*, 72(1):95–109, 1996.

[21] K. Pearson. Contributions to the mathematical theory of evolution. *Phil. Trans. R. Soc. Lond.*, 185:71–110, 1894.

[22] A. Pentland, K. Schwab, A. Marcus, J. Oyola, W. Hoffman, and M. Luzi. Personal data: The emergence of a new asset class. In *An Initiative of the World Economic Forum*, 2011.

[23] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, pages 452–461. AUAI Press, 2009.

[24] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, pages 811–820. ACM, 2010.

[25] M. T. Ribeiro, S. Singh, and C. Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *SIGKDD*, pages 1135–1144, New York, NY, USA, 2016. ACM.

[26] H. A. Sturges. The choice of a class interval. *JASA*, 21(153):65–66, 1926.

[27] X. Su and T. M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence*, 2009:4, 2009.

[28] P.-N. Tan et al. *Introduction to data mining.* Pearson Education India, 2006.

[29] M. Vescovi, C. Perentis, C. Leonardi, B. Lepri, and C. Moiso. My data store: toward user awareness and control on personal data. In *Ubicomp*, pages 179–182. ACM, 2014.

[30] P. Wang, J. Guo, and Y. Lan. Modeling retail transaction data for personalized shopping recommendation. In *CIKM*, pages 1979–1982. ACM, 2014.

[31] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng. Learning hierarchical representation model for nextbasket recommendation. In *SIGIR*, pages 403–412. ACM, 2015.

[32] F. Yu, Q. Liu, S. Wu, L. Wang, and T. Tan. A dynamic recurrent model for next basket recommendation. In *SIGIR*, pages 729–732. ACM, 2016.