Representing and Recommending Shopping Baskets with Complementarity, Compatibility, and Loyalty

Mengting Wan*, Di Wang†, Jie Liu†, Paul N. Bennett†, Julian McAuley*

*University of California, San Diego † Microsoft Corporation
*{m5wan,jmcauley}@ucsd.edu, †{wangdi, jie.liu, paul.n.bennett}@microsoft.com

ABSTRACT

We study the problem of representing and recommending products for grocery shopping. We carefully investigate grocery transaction data and observe three important patterns: products within the same basket complement each other in terms of functionality (*complementarity*); users tend to purchase products that match their preferences (*compatibility*); and a significant fraction of users repeatedly purchase the same products over time (*loyalty*). Unlike conventional e-commerce settings, *complementarity* and *loyalty* are particularly predominant in the grocery shopping domain. This motivates a new representation learning approach to leverage *complementarity* and *compatibility* holistically, as well as a new recommendation approach to explicitly account for users' 'must-buy' purchases in addition to their overall preferences and needs. Doing so not only improves product classification and recommendation performance on both public and proprietary transaction data covering various grocery store types, but also reveals interestingfindings about the relationships between preferences, necessity, and loyalty in consumer purchases.

ACM Reference Format:
Mengting Wan, Di Wang, Jie Liu, Paul N. Bennett, Julian McAuley. 2018. Representing and Recommending Shopping Baskets with Complementarity, Compatibility, and Loyalty. In *The 27th ACM International Conference on Information and Knowledge Management (CIKM '18), October 22–26, 2018, Torino, Italy*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3269206.3271786

1 INTRODUCTION

From online grocery shopping platforms (e.g. Amazon Fresh, Instacart) to automated checkout-free brick-and-mortar grocery stores (e.g. Amazon Go), recent technological innovations have enabled dramatic changes in people's grocery shopping experiences. As one of the most frequent shopping activities, vast amounts of grocery transaction data can be collected from multiple sources. On account of such innovations and volumes of data in this domain, various predictive tasks including personalized product recommendation, retail sales prediction, retail inventory optimization, and

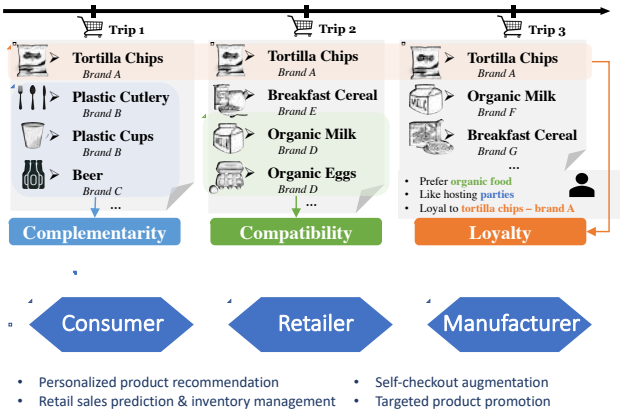


Figure 1: Three significant patterns observed in users' grocery baskets (item-to-item *complementarity*, user-to-item *compatibility*, and product *loyalty*) and their applications in the grocery industry.

personalized promotion strategies are worthy of interest. In order to facilitate these tasks, we seek to understand the semantics in users' grocery baskets, and to develop effective product representation and purchase prediction techniques.

Many modern and traditional recommendation techniques depend on learning latent representations of items from interaction data. A traditional example is a latent factor model [16], where a user-item interaction matrix is factorized by low-dimensional user and item terms. These methods in general attempt to recover the original interaction information globally, but may fail to capture subtle andfi ne-grained semantics of items. Inspired by word embedding techniques proposed for Natural Language Processing (NLP) tasks [21, 22, 24], recent item representation techniques have been developed for e-commerce [4, 12, 28, 29, 31]. In general these approaches are designed to learn representations which can effectively recover product co-occurrences either within a basket or across baskets from the same user. Foregoing handcrafted feature design, these methods automatically uncover useful (in terms of recommendation) representations of products. Both modern and traditional techniques have seen wide adoption in real-world e-commerce applications.

We hope to examine and adapt the above techniques to grocery transaction data. We notice that grocery shopping differs from conventional e-commerce applications, largely due to issues of *regularity* and *necessity* [23, 33]. Such nuances require us to carefully investigate grocery shopping behavior and build domain-specific

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/3269206.3271786

representations and recommendation algorithms. In particular, we consider the following patterns in users' grocery baskets:

- **Complementarity.** Users purchase multiple related products in the same basket to fulfill specific needs. These products complement each other in terms of functionality. For example, shoppers may buy plastic cups/forks/knives/plates and beer together for a party (Figure 1). Such item-to-item complementarity is critical as it not only captures products' latent functions but also reveals a user's intent in each basket.
- **Compatibility.** As with most e-commerce categories, compatibility between users' preferences and products' properties is paramount in grocery shopping. In addition, we notice that the above latent functions for complementarity may need to match users' preferences as well (e.g. plastic cups and cutlery are more likely to appear in a party lover's basket). This kind of cohesion inspires us to consider item-to-item complementarity and user-to-item compatibility holistically in our representation learning model for grocery shopping.
- **Loyalty.** A pronounced pattern in grocery transactions is that users tend to exhibit *'loyalty'* towards certain products, i.e., repeatedly purchasing the same product while rarely switching brands among alternatives. Such behavior is often contrary to the assumptions implicit in conventional recommendation models: typically if two products have similar representations and match a user's preferences (or needs), either could be recommended. However, if a user is loyal to one product, then its alternatives have systematically low probability of being purchased. In fact, simple user-wise product purchase frequency becomes a competitive baseline for recommending grocery products, as users' most loyal products can be 'memorized' based on these statistics. Of course, such a baseline lacks generalization power since it is unable to capture product semantics. Thus an appropriate algorithm for grocery recommendation should balance item-to-item complementarity, user-to-item compatibility, and users' product loyalty.

Our primary goal in this study is to leverage the above properties in the grocery shopping domain, and to develop a framework to understand the semantics of users' purchases. The representations we learn are generalizable and support tasks like automatic product categorization and personalized recommendation for grocery shopping at scale.

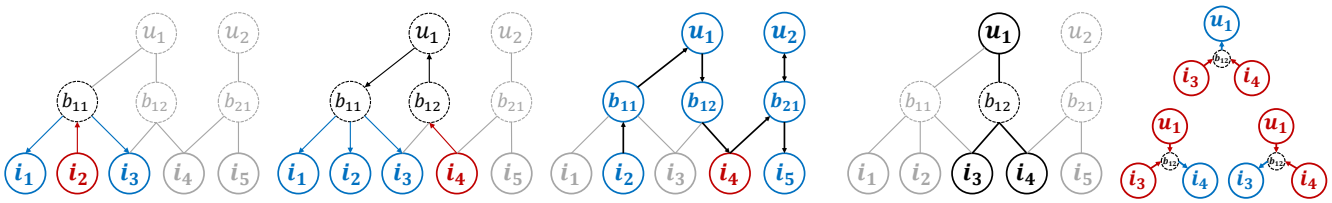
Contributions. In summary, our contributions are as follows:

- Inspired by the confluence of complementarity and compatibility, we focus on the core component in grocery transaction data—(*item*, *item*, *user*) triples linked by the same basket, i.e., two items purchased in the same basket from a user, and propose a representation learning model **triple2vec** to recover the above complementarity and compatibility holistically.
- On the basis of these product and user representations, we propose a novel algorithm **adaLoyal** for personalized grocery recommendation. Our method is capable of adaptively balancing users' "must-buy" products with preferences inferred from the low-dimensional representations.
- We conduct extensive experiments on two public and two proprietary datasets, which cover various grocery store types including conventional physical supermarkets, a convenience store, and an online grocery shopping platform.
- Based on the quantitative results from experiments, we demonstrate that the proposed **triple2vec** model is able to generate meaningful (in terms of product classification tasks) and useful (in terms of recommendation tasks) product representations. Particularly, by applying **adaLoyal**, performance of a variety of embedding learning methods can be dramatically improved. The effectiveness of product loyalty estimated from **adaLoyal** can be validated in our qualitative analysis as well.
- By conducting quantitative and qualitative analysis of our proposed methods, we reveal important and interesting insights about users' grocery shopping behaviors. The major insights include: 1) compared with cross-basket relationships, within-basket item-to-item complementarity is more useful in order to learn meaningfulfi ne-grained product representations; 2) modeling users' product loyalty and repeated purchases is critical in grocery product recommendation tasks, and such loyalty varies across different users, store types and product categories.

2 RELATED WORK

Traditional *model-based* item recommendation methods typically rely on Matrix Factorization (MF) techniques, e.g. via a latent factor model [16]. Of most relevance to grocery shopping are variants of MF for implicit feedback data where only positive signals (e.g. purchases) can be observed [13, 26]. MF-based methods have been extended to sequential recommendation (i.e., predicting items in a shopper's next basket, based on the context of their previous basket) by appropriately unifying MF and Markov Chains [27, 32]. More recently, by considering both user-to-item interactions and multiple associations among items simultaneously, such factorization techniques have been extended to within-basket recommendation (i.e., recommending products to be added to the current basket) [18]. In general, these models are optimized to directly favor global recommendation metrics which, while effective for recommendation, may fail to capture detailed semantics of products.

Recently, 'neural' representation learning methods including **word2vec** [21, 22] and **GloVe** [24] have achieved success on various NLP tasks. Particularly, the **skip-gram** technique [21, 22] has been widely extended to other domains including e-commerce [4, 12, 28, 29]. For example, **item2vec** was proposed by directly applying the **skip-gram** framework on itemsets, so that it can represent associations among products within the same itemset [4]. **prod2vec** and **bagged-prod2vec** were proposed to learn distributed product representations to support ad recommendations in *Yahoo! Mail* [12], where **skip-grams** are applied to recover product co-occurrence information across the same user's transactions. In order to overcome cold-start problems, several representation learning architectures have been developed to learn product embeddings by incorporating rich meta-data from different sources (e.g. product categories, images, description text) [28, 29, 34]. Moreover, a random-walk based network embedding method **metapath2vec** [10] was proposed to learn node representations from heterogeneous networks; as the relationships among users, baskets, and products can be easily represented as a heterogeneous graph, we consider this as a potential technique which can be applied on grocery shopping



(a) item2vec (b) prod2vec (c) metapath2vec (d) triple2vec

Figure 2: An illustrative example of different representation learning models. Here $\{u.\}$, $\{i.\}$, $\{b.\}$ are used to represent different users, items, and baskets. In each model, the given node is highlighted in red and the nodes for prediction are highlighted in blue.

Notation	Description
$\mathcal{U}$, $\mathcal{B}_u$, $\mathcal{I}_b$	user set, basket set for user u, item set of basket b
$\boldsymbol{f}_i$, $\boldsymbol{g}_i$, $\boldsymbol{h}_u$	two different embedding vectors for item i, and the embedding vector for user u
$s_{i,u}$, $s_{i,j,u}$	user u's preference score on item i, the cohesion score of the (*item*, *item*, *user*) triple (i, j, u)
$p_{i,u}$, $p_{i,uj}$	item purchase prob. given a user u, item purchase prob. given a user u and an item j; both are calculated from the original representation learning model
$\hat{p}^{(t)}_{i,u}$, $\hat{p}^{(t)}_{i,uj}$	item purchase prob. given a user u for transaction t, item purchase prob. given a user u and an item j for transaction t; both are generated from algorithm 1.
$q^{(t)}_{i,u}$, $l^{(t)}_{i,u}$	user u's empirical purchase frequency and estimated product loyalty of item i up to and including transaction t
$C^{(t)}_{i,u}$	whether product i is purchased by user u in the transaction t

Table 1: Notation.

transaction data. Details of some representative product embedding learning methods will be provided in Section 3.1. Note that these existing methods are designed for conventional e-commerce (or general graphs), which may fail to account for the strong cohesion of within-basket complementarity and compatibility exhibited in grocery purchases, nor do such methods explicitly account for repeated purchases.

On the other hand, brand loyalty and repeat purchasing behavior in grocery shopping have been theoretically and empirically studied in the areas of economics and psychology [9, 14, 25]. Although conceptually different, loyalty and repeat consumption correlate to each other [14], and such loyalty varies across products and categories [9, 25]. These studies also motivate us to extend the concept of 'loyalty' to large-scale product recommender systems. Recently, a 'Wide&Deep' approach was proposed to address both *memorization* and *generalization* issues in Google Play application recommendations [8]. Rather than modeling user-to-product loyalty, however, they seek to memorize frequently co-occurring products or features. Another line of relevant work includes modeling repeat consumption in online activities including video/music streaming [3, 5–7, 15, 17] and e-commerce [19]. The lifetime of an item in this context is relatively short and users' interest highly depends on recency, which is different from empiricalfi ndings from grocery shopping where the decline in loyalty is relatively small over time [9]. Therefore, new techniques need to be developed to handle the dynamics of this specific domain.

3 METHODS

Wefi rst briefly introduce the basic skip-gram-based embedding learning framework and several representative instantiations. Then we present the proposed representation learning method **triple2vec**, and show how to build downstream product classification and recommendation systems on these embeddings. Afterwards, we introduce **adaLoyal**, a recommendation algorithm which adaptively balances users' purchase frequency statistics and the generalization power from embedding learning methods. Important notation is included in Table 1.

3.1 Background

Several product representation learning methods are based on the skip-gram framework [22]. Essentially, they seek tofi nd item representations which are useful for predicting contextual (related) items or users, by defining different 'context windows.' In this section, we introduce them as different instantiations of a unified skip-gram framework on a heterogeneous graph (Figure 2), whose nodes are composed of different products, users, and baskets. Here we have two different types of links: 1) item-to-basket, indicating that an item is included in a basket, and 2) user-to-basket, indicating that a basket is purchased by a user. On this graph, several existing representation learning objectives can be cast as learning node representations which maximize the (log) likelihood of using a target node v to predict the contextual nodes C_v, i.e.,

$$\mathcal{L}_{sgn} = \sum_{v} \sum_{v' \in C_v} \log P(v'|v). \tag{1}$$

$P(v'|v)$ is commonly defined as $P(v'|v) = \frac{\exp(\boldsymbol{f}_v^T \boldsymbol{g}_{v'})}{\sum_{v''} \exp(\boldsymbol{f}_v^T \boldsymbol{g}_{v''})}$, where $\boldsymbol{f}_v$ and $\boldsymbol{g}_v$ are K dimensional 'input' and 'output' vector representations of a node.

We briefly introduce three representative methods of this type as follows:

- **item2vec.** Basket-level skip-grams can be directly applied on this graph, where we treat a particular item as a target node, and the rest of the products in the same basket as contextual nodes [4]. This definition relies on the assumption that products purchased in the same basket share similar semantics, which intuitively supports within-basket/"bundle" product recommendations. However, such co-purchase relationships may not be sufficient to capture personalized preferences toward products.

- **prod2vec.** Rather than directly applying Eq. (1) on baskets, in **prod2vec**, given a target product, the contextual nodes are defined as the products in recent baskets purchased by the same user [12].[1] Unlike the previous method which focuses on within-basket co-purchase relationships regardless of users, this approach emphasizes cross-basket item-to-item relationships for each user.
- **metapath2vec.** As mentioned, transaction logs can be transformed into a heterogeneous network. Therefore, a state-of-the-art network embedding learning method such as **metapath2vec** [10] can be applied here. In this scenario, we need to define a symmetric meta-path scheme: *item* → *basket* → *user* → *basket* → *item*, and generate different random walkers based on this predefined scheme. Specifically, we start with a random product, and sample a series of nodes to compose a random walker where each of the nodes consecutively links to the previous one on this meta-path. Then we select a given node, and define its surrounding nodes along the walk as 'contexts' C_v in Eq. (1). A concrete example is included in Figure 2c; here we sample a random walker $(i_2, b_{11}, u_1, b_{12}, i_4, b_{21}, u_2, b_{21}, i_5, \ldots)$, highlight a randomly selected target node i_4 in red and its surrounding nodes in blue.[2] Note that this *item* → *basket* → *user* → *basket* → *item* meta-path in general captures the semantics of products purchased by the same user but does not reflect product co-purchase relationships explicitly.

3.2 triple2vec: Representations from Triples

Unlike existing skip-gram-based product representations, we focus on the cohesion of each (*item*, *item*, *user*) reflecting two items purchased by the same user in the same basket. Specifically, the transaction logs for training consist of a series of such triples:

$$\mathcal{T} = \{(i, j, u) | i \in \mathcal{I}_b \wedge j \in \mathcal{I}_b \wedge i \neq j \wedge b \in \mathcal{B}_u \wedge u \in \mathcal{U}\}. \quad (2)$$

Then we define the cohesion score of each (i, j, u) triple as

$$s_{i,j,u} = \overbrace{\boldsymbol{f}_i^T \boldsymbol{g}_j}^{\text{item-to-item complementarity}} + \underbrace{\boldsymbol{f}_i^T \boldsymbol{h}_u + \boldsymbol{g}_j^T \boldsymbol{h}_u}_{\text{user-to-item compatibility}}, \quad (3)$$

where $\boldsymbol{f}_i, \boldsymbol{g}_j$ are two sets of representations for products and $\boldsymbol{h}_u$ represents the embedding vector of a user. Thefi rst term in Eq. (3) models the complementarity between two products within the same basket, i.e., whether two products exhibit similar semantics in terms of co-occurrence; the second and third terms are used to capture the compatibility between the product and the user, i.e., how well the product's (latent) properties match the user's preferences. A higher cohesion score indicates closer connections among the nodes in the triple. Finally, we aim to learn embeddings which optimize the occurrence likelihood of the training triples $\mathcal{T}$:

$$\mathcal{L} = \sum_{(i,j,u) \in \mathcal{T}} \big(\log P(i|j, u) + \log P(j|i, u) + \log P(u|i, j) \big), \quad (4)$$

where $P(i|j, u) = \frac{\exp(s_{i,j,u})}{\sum_{i'} \exp(s_{i',j,u})}$[3] and $P(u|i, j) = \frac{\exp(s_{i,j,u})}{\sum_{u'} \exp(s_{i,j,u'})}$. Essentially for each triple in $\mathcal{T}$, we iteratively 'knock out' a node and use the other two nodes to predict it. Figure 2 shows an illustrative example highlighting the difference between the proposed **triple2vec** and skip-gram-based models.

Negative Sampling. As with skip-gram-based models, a variation of Noise Contrastive Estimation (NCE) can be applied to approximate the softmax function in Eq. (4) and accelerate training [22]. For example, $\log P(i|j, u)$ in Eq. (4) can be replaced by

$$\log \sigma(s_{i,j,u}) + \sum \mathbb{E}_{i' \sim \mathcal{P}(i)} \log \sigma(-s_{i',j,u}), \quad (5)$$

where we sample N negative items from a pre-defined distribution $\mathcal{P}(i)$, and $\sigma(x) = \frac{1}{1+\exp(-x)}$. We achieve this through the NCE loss API provided in TensorFlow [2], where $\mathcal{P}(i)$ is defined as a log-uniform (Zipf) distribution. Specifically items are sorted in order of decreasing popularity and the probability of each item i being sampled is defined as $P(i) = \log \frac{r_i+2}{r_i+1}$, where r_i denotes the rank of item i. This negative sampling technique is used in all the representation learning methods implemented in our experiments, which empirically accelerates model convergence and improves quantitative performance compared to a uniform sampling strategy.

Representing and Recommending Products. Note that two sets of product embeddings $\boldsymbol{f}_i$ and $\boldsymbol{g}_i$ are learned from **triple2vec**. These embeddings describe the functions and properties of products from different angles, but the inner product between these two captures the cross product relationship—item-to-item complementarity. Therefore, for tasks to evaluate the semantics of products independently (e.g. product classification, competitor search), we follow the protocol in [24] and use the additive composition $\boldsymbol{f}_i + \boldsymbol{g}_i$ as the ultimate representation of each product i, which empirically gives a slight boost in most tasks. However, for predictive tasks where the cross-item relationship needs to be considered (e.g. item-to-item recommendation, complementary product search), we consider the inner product score $\boldsymbol{f}_i^T \boldsymbol{g}_j$ for two items i, j instead.

In particular, we consider two different recommendation scenarios: personalized next-basket product recommendation, and within-basket product recommendation:

- Given a user, when recommending products for the next basket, we replace the product embedding of the given product $\boldsymbol{g}_j$ by the average embedding of all the products in the user's previous baskets. Then we obtain a new preference score $s_{i,u}$ and the purchase probability is estimated as $p_{i,u} = \frac{\exp(s_{i,u})}{\sum_{i'} \exp(s_{i',u})}$.
- If products in the current basket are given, when recommending products to be added in the same basket, we replace $\boldsymbol{g}_j$ by the average embedding of all the products in the given item set.

Note that another set of preference scores could be obtained by exchanging $\boldsymbol{f}_i$ and $\boldsymbol{g}_i$. We also take the average of preference scores generated from these two methods and consider it as a third option. In our experiments, we report results from the method which yields the best validation performance.[4]

[1] We consider the bagged version of **prod2vec** here, where all products purchased in the same contextual basket need to be included together in the objective function.
[2] The neighborhood size is set to 8 in thisfi gure.
[3] Because of symmetry, by exchanging i and j, $P(j|i, u)$ can be obtained.
[4] In our experiments, this protocol is applied for all representation learning methods in which two heterogeneous embeddings are involved.

3.3 adaLoyal: Adaptively Updating and Estimating Product Loyalty

Wefi nd that in grocery baskets, a number of shoppers have their own 'must-buy' products. A preliminary analysis of 'must-buy' products is provided in Section 4.1 and Figure 3. Such repeat purchases could easily be measured based on user-wise item purchase frequency but may not be captured by low-dimensional product and user representations. Therefore, we introduce an algorithm **adaLoyal** to adaptively combine these two components and estimate users' product loyalty over time.

We start with a Bayesian view and then gradually build **adaLoyal** based on this principle. Specifically, for a user u, a product i and a transaction t, we have two predictive models for this basket: the prior purchase probability $p_{i,u}$, and the empirical item purchase frequency $q_{i,u}^{(t-1)}$ up to the given transaction t.[5] We introduce a latent loyalty indicator $L_{i,u}$, which acts as a 'switch' such that $L_{i,u}$ (u is loyal to i) causes predictions to be generated from frequency only, while $\neg L_{i,u}$ (u is not loyal to i) causes the probability to be estimated from representations. Then the ultimate item purchase probability can be generated from the following probabilistic mixture:

$$P(C_{i,u}^{(t)}=1) = P(L_{i,u}) \underbrace{P(C_{i,u}^{(t)}=1|L_{i,u})}_{\text{frequency model: } q_{i,u}^{(t-1)}} + P(\neg L_{i,u}) \underbrace{P(C_{i,u}^{(t)}=1|\neg L_{i,u})}_{\text{representation model: } p_{i,u}},$$

where $C_{i,u}^{(t)}=1$ indicates that i is purchased by user u in transaction t. On the other hand, given $C_{i,u}^{(t)}$, the posterior distribution of this loyalty indicator is:

$$P(L_{i,u}|C_{i,u}^{(t)}) = \frac{P(L_{i,u})\,P(C_{i,u}^{(t)}|L_{i,u})}{P(L_{i,u})\,P(C_{i,u}^{(t)}|L_{i,u}) + P(\neg L_{i,u})\,P(C_{i,u}^{(t)}|\neg L_{i,u})}. \quad (6)$$

Inspired by this posterior distribution, we seek to estimate a weight (i.e., the product 'loyalty') $l_{i,u}^{(t)} \in [0,1]$ which results in an ultimate product purchase probability for the next basket:

$$\tilde{p}_{i,u}^{(t+1)} = l_{i,u}^{(t)} q_{i,u}^{(t)} + (1-l_{i,u}^{(t)}) p_{i,u}. \quad (7)$$

i.e., $l_{i,u}^{(t)}$ is used to approximate $P(L_{i,u})$ adaptively. Finally we propose the **adaLoyal** algorithm as follows. We scan a user's transaction logs chronologically: 1) if a new product is observed, we activate its corresponding loyalty $l_{i,u}^{(t)}$ and set it to be a given initial value l_0; 2) if a product has been purchased before, $l_{i,u}^{(t)}$ is updated based on the posterior distribution of the loyalty indicator.

The pseudo-code of **adaLoyal** is given in algorithm 1 and the specific update rules for $l_{i,u}^{(t)}$ are provided in Eq. (8) and Eq. (9). In general, if a user starts repeatedly consuming a particular product beyond our expectation (i.e., it diverges from the representation model), the corresponding product loyalty will increase and its purchase frequency will be emphasized in thefi nal prediction.

Note that the same algorithm can also be applied to within-basket recommendation, where the ultimate product purchase probability for user u given product j in the basket can be estimated as $\tilde{p}_{i,uj}^{(t+1)} = l_{i,u}^{(t)} q_{i,u}^{(t)} + (1-l_{i,u}^{(t)}) p_{i,uj}$.

[5] $q_{i,u}^{(t-1)}$ is defined as the number of purchases of product i divided by the total number of products purchased up to the given transaction t.

Algorithm 1 Pseudo-code of **adaLoyal**

Input: $p_{i,u}$, $q_{i,u}^{(t)}$, $C_{i,u}^{(t)}$, l_0.
Output: $\tilde{p}_{i,u}^{(t)}$, $l_{i,u}^{(t)}$.
for each user u, each item i, each transaction t **do**
  if $q_{i,u}^{(t-1)}=0$ **then**
    // current item has not been purchased before
    assign $\tilde{p}_{i,u}^{(t)} = p_{i,u}$
    assign $l_{i,u}^{(t)} = l_0$, if $C_{i,u}^{(t)}=1$; $l_{i,u}^{(t)} = NA$, otherwise.
  else
    // loyalty of current item has been activated
    assign $\tilde{p}_{i,u}^{(t)} = l_{i,u}^{(t-1)} q_{i,u}^{(t-1)} + (1-l_{i,u}^{(t-1)}) p_{i,u}$
    if $C_{i,u}^{(t)}=1$ **then**
      assign $l_{i,u}^{(t)} = \frac{l_{i,u}^{(t-1)} q_{i,u}^{(t-1)}}{l_{i,u}^{(t-1)} q_{i,u}^{(t-1)} + (1-l_{i,u}^{(t-1)}) p_{i,u}}$ (8)
    else
      assign $l_{i,u}^{(t)} = \frac{l_{i,u}^{(t-1)} (1-q_{i,u}^{(t-1)})}{l_{i,u}^{(t-1)} (1-q_{i,u}^{(t-1)}) + (1-l_{i,u}^{(t-1)})(1-p_{i,u})}$ (9)
    end if
  end if
end for

4 EXPERIMENTS

We evaluate representations learned from **triple2vec** and the recommendation algorithm **adaLoyal** on two public and two proprietary grocery shopping transaction datasets. Code for the public datasets will be made available at publication time. In order to demonstrate that product embeddings are *meaningful* and *useful*, we evaluate 1) the item classification performance obtained with these representations; and 2) the accuracy of product recommendations obtained by leveraging these representations. Finally we evaluate 3) the boost in recommendation performance when using **adaLoyal** on top of these representations.

In addition to **triple2vec**, we consider the three methods described in Section 3.1: **item2vec** [4], **prod2vec** [12] and **metapath2vec** [10]. For all representation learning methods, we apply the same negative sampling approach, where the number of negative samples in Eq. (5) is set to 5.[6] **AdaGrad** [11], a stochastic gradient-based optimization method, is applied to learn all embeddings.

4.1 Datasets

We consider four real-world grocery transaction datasets, where *MSR-Grocery (WA)* and *MSR-Grocery (UT)* are two proprietary datasets collected from the Seattle and Salt Lake City areas (respectively). In order to ensure the reproducibility of our results, we also evaluate the performance of **triple2vec** and **adaLoyal** on two public datasets—*Dunnhumby* and *Instacart*.

- ***Dunnhumby.*** *The Complete Journey* dataset from Dunnhumby.[7] Transactions over two years collected from around two thousand households are included in this dataset. Users are frequent shoppers with an average shopping frequency of once per week.

[6] In practice, product bias terms are also added in Eq. (1) and Eq. (3) for all of these methods to capture overall item popularity.
[7] https://www.dunnhumby.com/sourcefiles

Dataset	#item	#user	#transaction	#trans./#user	basket size	#department	#dept. ≥ 5	#category	#cat. ≥ 5
Dunnhumby	26,780	2,500	269,974	107.99	9.02	31	24	310	255
Instacart	42,987	206,209	3,345,786	16.23	10.10	21	21	134	134
Grocery(WA)	16,497	47,939	360,222	7.51	4.81	34	26	306	177
Grocery(UT)	26,821	60,421	634,733	10.51	9.80	24	22	288	247

Table 2: Basic dataset statistics.

- ***Instacart.*** This dataset was published by *instacart.com* [1], a web service that provides same-day grocery delivery in the US. It contains over 3 million grocery orders from more than 200 thousand users. The specific date of each order is missing but the sequence order of transactions by each user is provided.
- ***MSR-Grocery (WA).*** This dataset is collected from a single convenience store in the Seattle area and wasfi rst used in [30]. We extend this dataset to include 12 months of transactions from around 360 thousand users. Because of the type of this store, users tend to have fewer transactions and smaller basket sizes compared with other datasets.
- ***MSR-Grocery (UT).*** Finally we collected 8 months of transactions from two mid-size grocery stores in the Salt Lake City area. These two stores are from the same grocery chain and include relatively diverse consumers including households and college students.

After removing rare products (fewer than 10 purchases) from these datasets, the basic statistics of these preprocessed datasets are listed in Table 2. The following rules are applied to split transaction data into train/validation/test sets: 1) for users who have more than one transaction, their most recent transaction is used for testing; 2) for users who have more than two transactions, their second-to-last transactions is used for validation; 3) all the other transactions are used for training. All embedding learning and recommendation models are learned on the training data, and all hyper-parameters are selected based on validation performance. All recommendation results are reported on the held-out test data.

As a preliminary analysis of users' product loyalty in grocery shopping, we explore the distribution of the item purchase frequency of each user's most favored product, $q_{\max}$.[8] In Figure 3 we split users into three groups: $q_{\max} \in (0, 0.1], (0.1, 0.5], (0.5, 1]$ to show the distribution. We notice that the fraction of users where $q_{\max} \leq 0.1$ is limited, which means most users exhibit some repeat consumptions. Moreover, different from other datasets, around 70% of users in *Instacart* have products which are repeatedly purchased in more than half of their transactions. A possible explanation could be that *Instacart* is collected from regular shoppers on an online grocery shopping platform, where users may repeatedly seek the same products for efficiency rather than browsing and exploring as in physical stores.

4.2 Product Classification

Hierarchical product categories are provided in all four datasets. We treat the top-level hierarchy as the 'department' and the second-level as the 'category,' and remove small departments and categories with fewer than 5 products (see Table 2). We evaluate the quality of the product embeddings learned by different methods over both coarse-grained (department) andfi ne-grained (category) classification. In particular, we apply a one-vs-all linear logistic regression classifier on the product embeddings, where the hyper-parameter for the l_2 regularizer is selected based on a 5-fold cross-validation.

[8]For users who have at least 10 transactions in each dataset, we calculate the user-wise item purchase frequency andfi nd the maximum $q_{\max}$ for each user (i.e., the purchase frequency of the user's most favored product).

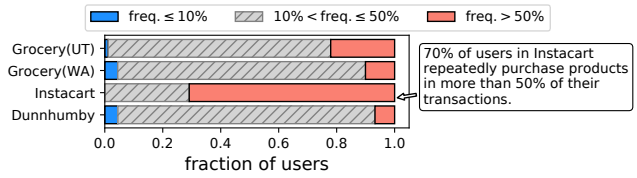


Figure 3: Distribution of the purchase frequency of each user's most favorite product.

	Dunnhumby		*Instacart*		*Grocery(WA)*		*Grocery(UT)*	
Method	micro	macro	micro	macro	micro	macro	micro	macro
item2vec	0.665	0.108	0.377	0.283	*0.608*	0.345	0.620	0.239
prod2vec	0.617	0.066	0.330	0.218	0.480	0.212	0.491	0.093
m.2vec	0.627	0.071	0.331	0.221	0.441	0.144	0.484	0.067
triple2vec	*0.669*	*0.114*	*0.382*	*0.294*	0.581	*0.361*	*0.623*	*0.293*

(a) F1 metrics on coarse-grained (department) classification

	Dunnhumby		*Instacart*		*Grocery(WA)*		*Grocery(UT)*	
Method	micro	macro	micro	macro	micro	macro	micro	macro
item2vec	0.160	0.046	0.187	0.075	0.518	*0.010*	0.275	0.094
prod2vec	0.087	0.015	0.106	0.030	0.518	0.009	0.119	0.023
m.2vec	0.078	0.007	0.155	0.036	0.518	0.007	0.091	0.008
triple2vec	*0.175*	*0.049*	*0.189*	*0.082*	*0.519*	*0.010*	*0.291*	*0.097*

(b) F1 metrics onfi ne-grained (category) classification

Table 3: Detailed results on product classification tasks ($K = 32, r = 50\%$, the best performance is underlined). All reported improvements are significant at 1%.

Results. Wefi x the embedding dimensionality to $K = 32$ and use half of the products for training and the other half for testing (i.e., label fraction $r = 0.5$). Then we repeat each experiment 10 times, and report the average micro-F1 and macro-F1 scores in Table 3.[9] We also vary the embedding dimension K, label fraction r, and report the classification results in Figures 5a and 5b on the *Dunnhumby* dataset to address the sensitivities of these hyper-parameters [10].

[9]All differences are significant at 5% level.

[10]$K \in \{8, 16, 32, 64, 128\}$, $r \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$

Wefi nd that **triple2vec** substantially and consistently outperforms all baselines on both department and category classification. The improvement is increased with larger embedding dimension K. The non-personalized method **item2vec** in general yields better classification results compared with the other two baselines. This may indicate that in order to learn meaningful grocery product representations, within-basket item-to-item complementarity is relatively more significant compared with cross-basket item-to-item relationships or item-to-user relationships. Note that in Table 3, the performance of **item2vec** is particularly strong on the convenience store dataset (*MSR-Grocery (WA)*), which may reveal the particular significance of such item-to-item complementarity in this type of stores.

4.3 Personalized Recommendation

In addition to product classification tasks, we evaluate the product/user representations and the performance of the proposed **adaLoyal** algorithm on two recommendation tasks: next-basket recommendation and within-basket recommendation. Particularly, we consider the original purchase probability estimated based on embeddings learned from **item2vec**, **prod2vec** and **metapath2vec** as baselines; **adaLoyal** can be applied on top of any of these. For both recommendation tasks, as in [26, 27], we rank products based on the predicted purchase probability, and consider the Area Under the ROC Curve (AUC) as an overall ranking metric, and Normalized Discounted Cumulative Gain (NDCG) as a top-biased evaluation metric. In this section, we report detailed results with a fixed dimension of item/user embeddings $K = 32$ (Table 4) and vary it on the *Dunnhumby* dataset for sensitivity analysis (Figure 5c). The initial loyalty value l_0 is selected based on the performance on the validation set.[11]

Next-Basket Recommendation. Wefi rst consider recommending products for users' next baskets. The same prediction method in Section 3.2 is applied for all representation learning baselines. If a user embedding is not available, we use the average embedding of items in a user's training baskets instead. We further consider two additional baselines: overall item purchase frequency (**itemPop**) and user-wise item purchase frequency (user-wise **itemPop**).

As next-basket recommendation is a classic recommendation task, we consider two state-of-the-art *supervised* implicit-feedback recommendation baselines as well: 1) **BPR-MF** [26], an item recommendation model which factorizes the user-item compatibility by approximately optimizing the AUC ranking metric, and 2) **FPMC** [27], where sequential information (via afi rst-order Markov Chain) is considered in addition to user-to-item compatibility. To make a fair comparison, we extract item and user embeddings from all representation learning methods and learn a weighted inner product between these two kinds of embeddings as a ranking score. The associated weights are learned by applying the same pairwise ranking loss (i.e., the **BPR** loss [26]). Note we adopt the same supervised learning protocol here but only need to learn K parameters,[12] where K is the dimensionality of latent embeddings.

Detailed results on this task are reported in Table 4a. In general **triple2vec**+**adaLoyal** and **metapath2vec**+**adaLoyal** outperform other embedding learning methods, as both explicitly model user-item compatibility during the representation learning process. We notice that although **BPR-MF** and **FPMC** outperform most representation learning methods without incorporating product loyalty, by applying the same supervised learning loss function and learning minimal parameters, these representation learning methods can achieve comparable results on some datasets in terms of AUC and better top-biased ranking performance on all the datasets in terms of NDCG (see the difference between 'BPR-MF' and '+BPR' in Table 4a). Note that user-wise **itemPop** yields strong performance based on NDCG but poor performance based on the overall ranking metric AUC, as such a frequency-based method is not capable of capturing basket semantics and lacks generalization power for future purchases. Nevertheless, its top-biased performance is

	Dunnhumby		*Instacart*		*Grocery(WA)*		*Grocery(UT)*	
Method	AUC	NDCG	AUC	NDCG	AUC	NDCG	AUC	NDCG
itemPop	0.799	0.129	0.918	0.145	0.809	0.130	0.854	0.137
+userwise	0.732	0.175	0.773	0.273	0.591	0.150	0.607	0.141
BPR-MF	0.861	0.136	0.964	0.161	*0.831*	0.136	0.862	0.139
FPMC	0.853	0.137	0.963	0.163	0.821	0.139	0.865	0.139
item2vec	0.801	0.132	0.945	0.111	0.794	0.138	0.822	0.108
+BPR	0.851	0.145	0.964	0.188	0.804	0.141	0.846	0.138
+adaLoyal	0.853	0.181	0.963	0.270	0.805	0.174	0.858	0.133
prod2vec	0.796	0.119	0.945	0.115	0.790	0.137	0.826	0.119
+BPR	0.850	0.144	0.964	0.183	0.807	0.144	0.852	0.140
+adaLoyal	0.848	0.154	0.964	0.273	0.803	0.175	0.853	0.138
m.2vec	0.838	0.144	0.954	0.125	0.810	0.126	0.846	0.113
+BPR	0.854	0.153	0.959	0.189	0.809	0.145	0.856	0.147
+adaLoyal	0.862	*0.182*	0.967	0.269	0.820	0.174	0.874	0.149
triple2vec	0.852	0.129	0.959	0.128	0.817	0.137	0.848	0.124
+BPR	0.861	0.142	0.962	0.186	0.819	0.149	0.854	0.144
+adaLoyal	*0.870*	0.166	*0.968*	*0.277*	0.830	*0.176*	*0.875*	*0.152*

(a) AUC and NDCG on next-basket recommendation.

	Dunnhumby		*Instacart*		*Grocery(WA)*		*Grocery(UT)*	
Method	AUC	NDCG	AUC	NDCG	AUC	NDCG	AUC	NDCG
itemPop	0.795	0.129	0.918	0.145	0.809	0.131	0.854	0.137
+userwise	0.730	0.174	0.773	0.272	0.590	0.149	0.606	0.141
item2vec	0.831	0.145	0.941	0.116	0.835	0.159	0.868	0.117
+adaLoyal	0.878	0.183	0.965	0.273	*0.849*	0.190	0.883	0.140
prod2vec	0.803	0.121	0.941	0.125	0.820	0.148	0.850	0.125
+adaLoyal	0.856	0.173	0.965	0.273	0.836	0.184	0.866	0.142
m.2vec	0.834	0.144	0.944	0.125	0.810	0.126	0.846	0.113
+adaLoyal	0.858	0.182	0.960	0.269	0.820	0.173	0.874	0.146
triple2vec	0.864	0.145	0.960	0.127	0.830	0.153	0.869	0.132
+adaLoyal	*0.879*	*0.185*	*0.970*	*0.279*	0.843	*0.191*	*0.885*	*0.157*

(b) AUC and NDCG on within-basket recommendation.

Table 4: Detailed results on recommendation tasks ($K = 32$).

[11] $l_0 \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$

[12] In practice, we may need to learn item bias terms as well.

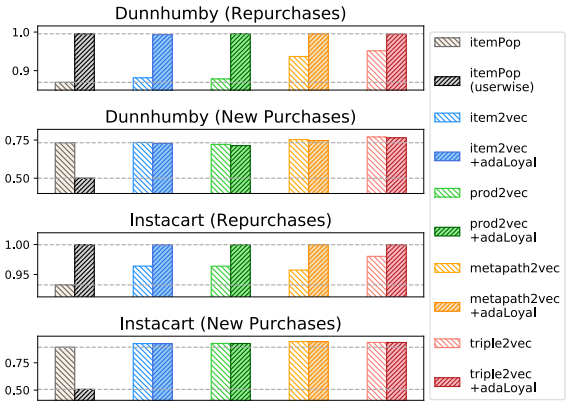


Figure 4: Results for repurchased products and newly purchased products in next-basket recommendation tasks on the *Dunnhumby* and *Instacart* datasets (in terms of AUC).

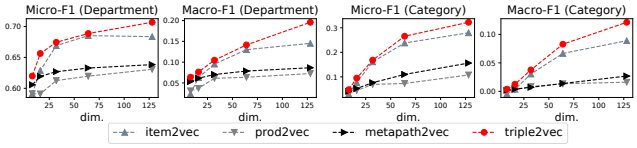
stronger than **BPR-MF** and **FPMC** which directly optimize a ranking metric. A possible reason could be that users' favoritism to some products is difficult to model using latent low-dimensional representations but easy to memorize based on purchase frequency. For the same reason, the performance of all embedding learning methods is significantly boosted in terms of both AUC and NDCG by applying **adaLoyal** to effectively combine these two models (see the difference between thefi rst row and '+adaLoyal' in each group of Table 4a).

We further explore the improvement from **adaLoyal** on users' repurchases (i.e., products that have be purchased in the given user's training transactions) and new purchases (i.e., products that have not been purchased by the user before the test transaction). Results on the ***Dunnhumby*** and ***Instacart*** datasets are provided in Figure 4. Wefi nd that by applying **adaLoyal**, recommendations on repurchases will be boosted to the upper bound provided by user-wise **itemPop**. Doing so only sacrifices limited performance when generalizing to new purchases. This implies that the algorithm benefits from both the frequency model and universal embeddings by successfully distinguishing 'must-buy' and 'on-demand' products. The effectiveness of estimated product loyalties will be further validated in the subsequent case studies.

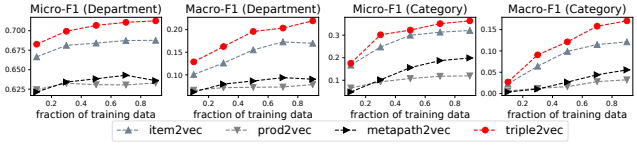
Within-Basket Recommendation. Next we consider a setting where we assume some products in the basket are given and we recommend 'complimentary' products to be added to the basket. Specifically, for a transaction which contains more than one item, we assume half of the products are given and predict the remaining half. For **metapath2vec**, **itemPop** and user-wise **itemPop**, we directly apply next-basket predictions as they do not explicitly account for item-to-item relationships. For **item2vec** and **prod2vec**, we use the item-to-item complementarity score for preference prediction (i.e., $p_{i,uj} \propto \exp(\boldsymbol{f}_i^T \boldsymbol{g}_j)$).

Detailed results are included in Table 4b and **triple2vec** still dominates other methods in most cases. Besides, we notice that **item2vec** becomes a competitive method in this scenario and outperforms other baselines. This pattern can be observed when experimenting with different numbers of embedding dimensions as well

(a) Dimension K (Department and Category Classification, $r = 0.5$)



(b) Label Fraction r (Department and Category Classification, $K = 128$)



(c) Dimension K (Next-Basket and Within-Basket Recommendation)

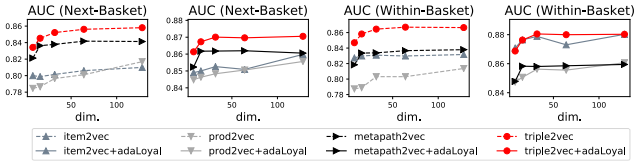


Figure 5: Sensitivity analysis in product classification and recommendation tasks on the *Dunnhumby* dataset.

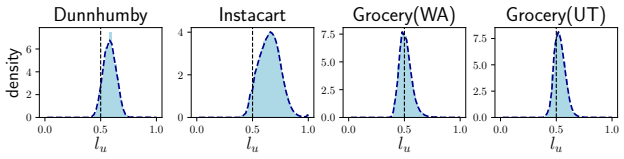


Figure 6: Histograms of user's product loyalty across different datasets, where l_u represents the average product loyalty of each user with the same initialization $l_0 = 0.5$.

(see Figure 5c). This suggests that by explicitly including within-basket item-to-item interactions in the preference score, item-to-item complementarity can be captured and thus improve within-basket recommendations.

4.4 Case Studies

In addition to our quantitative results from product classification and recommendation, we conduct three case studies on our largest public dataset, *Instacart*, to demonstrate the effectiveness of the product embeddings, and to explore the variety of product loyalty across different users and categories.

Loyalty Analysis. Here we explore product loyalty estimated from **adaLoyal+triple2vec** in detail. Wefi x the initial loyalty $l_0 = 0.5$, and collect the estimated loyalties $l_{i,u}^{(t)}$ for each user at thefinal timestamp in the training set. Then we calculate the average value of product loyalties for each user and provide its distribution across different datasets in Figure 6. In thisfi gure, wefi nd that *Instacart* is more "loyalty"-dominated compared with physical grocery stores.

We also provide concrete transaction examples in Table 5 to illustrate how users behave differently from each other in terms

User A ($l_u = 1.00$)	**User B** ($l_u = 0.57$)	**User C** ($l_u = 0.37$)
Sparkling Water, Bottles	Spinach Artichoke Dip, Taboule Salad, ...	Olive Oil Soap, Citrus Castile Soap, Peppermint Castile Soap ...
Sparkling Water, Bottles	Packaged Grape Tomatoes	Coconut Chips – Sea Salt, Coconut Chips – Original ...
Sparkling Water, Bottles	Bag of Organic Bananas, Taboule Salad	Compostable Forks
Sparkling Water, Bottles	Fuji Apples, Seedless Cucumbers, ...	Grunge Buster Grout And Tile Brush
Sparkling Water, Bottles	Bag of Organic Bananas, Sweet Kale Salad Mix	Pumpkin Seed Cheddar Crispbreads, Seedlander Crispbreads
Sparkling Water, Bottles	Spinach Artichoke Dip, Seedless Red Grapes, ...	Zinc Target Mins 50 Mg Gluten Free Tablets

Table 5: Baskets from users with the same number of transactions, but different average product loyalties in the *Instacart* dataset.

loyal dept.	loyalty	unloyal dept.	loyalty	loyal cat.	loyalty	unloyal cat.	loyalty
pets	0.64	pantry	0.52	milk	0.68	kitchen supp.	0.45
dairy/eggs	0.63	personal care	0.52	eggs	0.68	baking decor.	0.45
beverages	0.61	other	0.52	water/seltzer	0.66	spices	0.46
bakery	0.61	household	0.53	energy drinks	0.65fi	rst aid	0.46
breakfast	0.61	international	0.54	lactose free	0.65	beauty	0.48

(a) Department/category ranking based on the raw loyalty score l_u.

loyal dept.	z_l	z_f	$z_l - z_f$	unloyal dept.	z_l	z_f	$z_l - z_f$
pets	1.638	-0.826	2.464	meat/seafood	0.326	2.353	-2.027
dairy/eggs	1.314	-0.734	2.048	international	-1.117	0.857	-1.973
bulk	-0.395	-1.606	1.210	pantry	-1.768	-0.223	-1.544
babies	0.509	-0.606	1.115	beverages	0.951	1.499	-0.547
alcohol	0.100	-0.717	0.817	dry goods/pasta	-0.349	0.165	-0.514
loyal cat.	z_l	z_f	$z_l - z_f$	unloyal cat.	z_l	z_f	$z_l - z_f$
mint gum	0.933	-0.356	1.289	fresh fruit	0.408	4.817	-4.408
dog food care	1.215	0.067	1.148	fresh vegetables	0.255	4.234	-3.980
granola	0.986	-0.158	1.144	pack. veg./fruits	0.525	3.006	-2.481
energy drinks	1.709	0.656	1.053	spices	-2.377	-0.618	-1.759
eggs	2.264	1.213	1.051	fresh herbs	-0.932	0.464	-1.396

(b) Department/category ranking based on the relative score $z_l - z_f$ (i.e., difference between z-scores of loyalty and frequency).

Table 6: Thefi ve most loyal/unloyal departments and categories in the *Instacart* dataset.

of product loyalty. Wefi nd a few users who exhibit strong product loyalty similar to *User A* in Table 5, i.e., they order certain products in every transaction. However, most users' shopping patterns are better reflected by *User B*'s transactions, where they have strong preferences on some products but occasionally switch brands (e.g. Taboule Salad and Sweet Kale Salad Mix). Different from these two kinds of users, product-*unloyal* consumers prefer to explore the store rather than sticking to particular products. For example, *User C* in Table 5 has hardly any repeated product consumptions, rather they buy different products with the same function (e.g. various soaps, coconut chips, crispbreads) in the same basket.

Next we calculate the average product loyalties for each department and each category. Based on these statistics, thefi ve most loyal and unloyal departments/categories are listed in Table 6a. Note that the distribution of this absolute loyalty estimation potentially correlates to the demand/necessity of a category. For example, we observe users are loyal to products such as "milk" and "eggs" (i.e., they repeatedly purchase a specific product in these categories) while these categories are highly in-demand and need to be purchased frequently in our daily life. To further investigate the loyalty/necessity relationship we normalize both product loyalties and

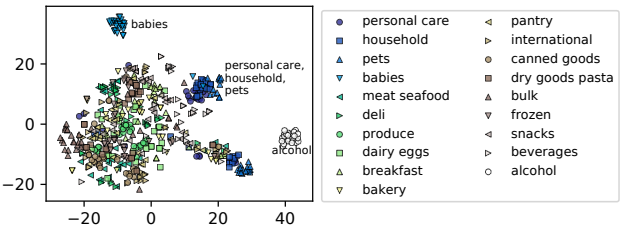


Figure 7: 2d t-SNE projections of the 32-dimensional product embeddings learned from triple2vec on the *Instacart* dataset.

department/category purchase frequencies into z-scores.[13] Then we calculate the difference between their z-scores to investigate how surprisingly loyal or unloyal is a particular department/category. Based on this difference, we provide thefi ve most relatively loyal and unloyal departments/categories in Table 6b, where we observe that users are relatively unloyal to fresh fruits, vegetables, meat, and spices but loyal to mint gums, granola, baby products, alcohol, and pet products.

Similarity Search. Next we demonstrate the effectiveness of our product embeddings by showing how they can be used to search for 'complementary' and 'substitutable' products. We calculate the complementarity score (i.e., $\boldsymbol{f}_i^T \boldsymbol{g}_j$) between all products and the given query product, in order to retrieve the topfi ve complements. For substitutable products, we use the additive composition $\boldsymbol{f}_i + \boldsymbol{g}_i$, and retrieve the topfi ve products (essentially i's competitors) based on cosine similarity. In Table 7 we query the two most popular products "Banana" and "Organic Banana." We observe that milk, yogurt, and granola are likely to be complements for bananas while other fresh fruits can be regarded as similar products/competitors. Another interesting pattern is that most retrieved products for "Organic Banana" are organic products while none are for the (non-organic) "Banana." This indicates that our representation learning method **triple2vec** can capture latent properties (e.g. "organic") of products, which might be particularly useful when recommending products to match users'fi ne-grained preferences.

Product Visualization. For each department in *Instacart*, we select the 30 most popular products and visualize the low-dimensional product representations learned from **triple2vec** in Figure 7 via t-SNE [20]. In thisfi gure, we notice that **triple2vec** automatically

[13] The z-score is defined as: $(x - mean(x))/sd(x)$

Product Query: "Banana"

Complements	Score	Competitors	Score
Whole Milk With Vitamin D	3.46	Fuji Apple	0.97
Plain Yogurt	3.11	Honeycrisp Apple	0.96
Apple Blueberry Granola	3.06	Cucumber Kirby	0.93
Orange Navel	3.01	Large Lemon	0.92
Milk Chocolate Nutrition Shake	2.99	Large Grapefruit	0.92

Product Query: "Organic Banana"

Complements	Score	Competitors	Score
Organic Papaya	3.72	Organic Strawberries	0.96
Organic 2% Milk	3.69	Organic Raspberries	0.94
Carbonated Water	3.66	Organic Blueberries	0.94
Organic Bosc Pears	3.61	Organic Hass Avocado	0.93
Organic Applesauce	3.55	Organic Large Extra Fuji Apple	0.92

Table 7: Complement and competitor search for "Banana" and "Organic Banana" in the *Instacart* dataset. Note the z-normalized complementarity score and the cosine similarity score are shown in the second and last columns.

organizes these products around different functions. For example, products in the personal care, household and pets departments are separated from food products (e.g. produce, meat/seafood, dairy/eggs). In addition, two relatively isolated departments – *babies* and *alcohol* can be observed in this dataset. Note these two are relatively loyal but infrequently purchased departments (see Table 6b), which may reflect users' unique shopping patterns. Purchases of baby products are normally necessities, which results in relatively tight connections. Similarly, alcohol products are different from other daily food consumptions in nature. Also, purchasing such products usually incurs additional 'costs' to users (e.g. providing valid identification). Therefore, users' shopping patterns in this department are dramatically different from others.

5 CONCLUSIONS

We investigated grocery shopping behavior and observed three important patterns in users' baskets—complementarity between products, compatibility between users and products, and users' product loyalty. We proposed a new representation learning method, **triple2vec**, to holistically leverage complementarity and compatibility, and designed a novel algorithm **adaLoyal** for product recommendation by adaptively balancing universal product embeddings and users' product loyalty over time. We demonstrated their effectiveness through quantitative and qualitative results on two public and two proprietary grocery datasets.

The idea of complementarity, compatibility, and loyalty is not limited to grocery shopping but can be widely applied on other domains, especially those with repeated consumptions (e.g. music streaming). It would also be interesting to extend **adaLoyal** to be a Bayesian reinforcement learning framework, which could learn the product loyalty and update the recommendations in an interactive environment.

REFERENCES

[1] The instacart online grocery shopping dataset 2017. Accessed on Dec. 2017. https://www.instacart.com/datasets/grocery-shopping-2017.

[2] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[3] A. Anderson, R. Kumar, A. Tomkins, and S. Vassilvitskii. The dynamics of repeat consumption. In *WWW*, 2014.

[4] O. Barkan and N. Koenigstein. Item2vec: Neural item embedding for collaborative filtering. In *Workshop on Machine Learning for Signal Processing*, 2016.

[5] A. R. Benson, R. Kumar, and A. Tomkins. Modeling user consumption sequences. In *WWW*, 2016.

[6] J. Chen, C. Wang, and J. Wang. Modeling the interest-forgetting curve for music recommendation. In *MM*, 2014.

[7] J. Chen, C. Wang, and J. Wang. A personalized interest-forgetting markov model for recommendations. In *AAAI*, 2015.

[8] H.-T. Cheng, L. Koc, J. Harmsen, T. Shaked, T. Chandra, H. Aradhye, G. Anderson, G. Corrado, W. Chai, M. Ispir, et al. Wide & deep learning for recommender systems. In *1st Workshop on Deep Learning for Recommender Systems*, 2016.

[9] J. Dawes, L. Meyer-Waarden, and C. Driesener. Has brand loyalty declined? a longitudinal analysis of repeat purchase behavior in the uk and the usa. *Journal of Business Research*, 68(2):425–432, 2015.

[10] Y. Dong, N. V. Chawla, and A. Swami. metapath2vec: Scalable representation learning for heterogeneous networks. In *KDD*, 2017.

[11] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *JMLR*, 2011.

[12] M. Grbovic, V. Radosavljevic, N. Djuric, N. Bhamidipati, J. Savla, V. Bhagwan, and D. Sharp. E-commerce in your inbox: Product recommendations at scale. In *KDD*, 2015.

[13] Y. Hu, Y. Koren, and C. Volinsky. Collaborativefi ltering for implicit feedback datasets. In *ICDM*, 2008.

[14] J. Jacoby and D. B. Kyner. Brand loyalty vs. repeat purchasing behavior. *Journal of Marketing research*, pages 1–9, 1973.

[15] K. Kapoor, K. Subbian, J. Srivastava, and P. Schrater. Just in time recommendations: Modeling the dynamics of boredom in activity streams. In *WSDM*, 2015.

[16] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *IEEE Computer*, 2009.

[17] D. Kotzias, M. Lichman, and P. Smyth. Predicting consumption patterns with repeated and novel events. *TKDE*, 2018.

[18] D. T. Le, H. W. Lauw, and Y. Fang. Basket-sensitive personalized item recommendation. In *IJCAI*, 2017.

[19] L. Lerche, D. Jannach, and M. Ludewig. On the value of reminders within e-commerce recommendations. In *UMAP*, 2016.

[20] L. v. d. Maaten and G. Hinton. Visualizing data using t-SNE. *JMLR*, 2008.

[21] T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[22] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013.

[23] S. J. Moses and L. D. Babu. Buyagain grocery recommender algorithm for online shopping of grocery and gourmet foods. *International Journal of Web Services Research (IJWSR)*, 2018.

[24] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014.

[25] P. Quester and A. Lin Lim. Product involvement/brand loyalty: is there a link? *Journal of product & brand management*, 12(1):22–38, 2003.

[26] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback. In *UAI*, 2009.

[27] S. Rendle, C. Freudenthaler, and L. Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *WWW*, 2010.

[28] Z. Sun, J. Yang, J. Zhang, A. Bozzon, Y. Chen, and C. Xu. MRLR: multi-level representation learning for personalized ranking in recommendation. In *IJCAI*, 2017.

[29] F. Vasile, E. Smirnova, and A. Conneau. Meta-prod2vec: Product embeddings using side-information for recommendation. In *RecSys*, 2016.

[30] M. Wan, D. Wang, M. Goldman, M. Taddy, J. Rao, J. Liu, D. Lymberopoulos, and J. McAuley. Modeling consumer preferences and price sensitivities from large-scale grocery shopping transaction logs. In *WWW*, 2017.

[31] J. Wang, P. Huang, H. Zhao, Z. Zhang, B. Zhao, and D. L. Lee. Billion-scale commodity embedding for e-commerce recommendation in alibaba. *arXiv preprint arXiv:1803.02349*, 2018.

[32] P. Wang, J. Guo, Y. Lan, J. Xu, S. Wan, and X. Cheng. Learning hierarchical representation model for next-basket recommendation. In *SIGIR*, 2015.

[33] M. Yuan, Y. Pavlidis, M. Jain, and K. Caster. Walmart online grocery personalization: Behavioral insights and basket recommendations. In *ER*, 2016.

[34] Y. Zhang, Q. Ai, X. Chen, and B. Croft. Joint representation learning for top-n recommendation with heterogeneous information sources. In *CIKM*, 2017.