Tensorized Determinantal Point Processes for Recommendation

Romain Warlop
romain@fifty-five.com
fifty-five, Paris, France
SequeL Team, Inria Lille, France

Jérémie Mary
j.mary@criteo.com
Criteo AI Lab
Paris, France

Mike Gartrell
m.gartrell@criteo.com
Criteo AI Lab
Paris, France

ABSTRACT

Interest in determinantal point processes (DPPs) is increasing in machine learning due to their ability to provide an elegant parametric model over combinatorial sets. In particular, the number of required parameters in a DPP grows only quadratically with the size of the ground set (e.g., item catalog), while the number of possible sets of items grows exponentially. Recent work has shown that DPPs can be effective models for product recommendation and basket completion tasks, since they are able to account for both the diversity and quality of items within a set. We present an enhanced DPP model that is specialized for the task of basket completion, the tensorized DPP. We leverage ideas from tensor factorization in order to customize the model for the next-item basket completion task, where the next item is captured in an extra dimension of the model. We evaluate our model on several real-world datasets, and find that the tensorized DPP provides significantly better predictive quality in several settings than a number of state-of-the art models.

KEYWORDS

Determinantal Point Process, basket completion, Tensor

ACM Reference Format:
Romain Warlop, Jérémie Mary, and Mike Gartrell. 2019. Tensorized Determinantal Point Processes for Recommendation. In *The 25th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '19), August 4–8, 2019, Anchorage, AK, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3292500.3330952

1 INTRODUCTION

Increasing the number of items in the average shopping basket is a major concern for online retailers. While there are a wide range of possibles strategies, this work focuses on the algorithm responsible for proposing a set of items that is best suited to completing the user's current shopping basket.

Basket analysis and completion is a very old task for machine learning. For many years association rule mining [2] has been the state-of-the-art. Even though there are different variants of this algorithm, the main principle involves computing the conditional probability of buying an additional product by counting co-occurrences in past observations. Due to computational cost and robustness, modern approaches favor item-to-item collaborative filtering [19], or using logistic regression to predict if a user will purchase an item based on binary purchase scores obtained from shopping baskets [18].

As reported in the *Related Work* section, standard collaborative filtering approaches need to be extended to correctly capture diversity among products. Indeed, in basket completion, some form of diversity needs to be incorporated, since recommending an item that is too similar to other items in the basket will not be relevant to the user. Practitioners often mitigate this problem by adding constraints to the recommended set of items. As an example, when using categorical information, it is possible to force the recommendation of a pair of matching shoes when trousers are added to the basket, even if natural co-sale patterns would lead to the recommendation of other trousers. In this situation the presence of diversity in the recommendations is not directly driven by the learning algorithm, but by side information and expert knowledge. Ref. [28] proposes an effective Bayesian method for learning the weights of the categories in the case of visual search when categories are known.

However, it may be more interesting to directly learn the appropriate diversity without relying on extra information. Naive learning of diversity directly from the data without using side information comes at a high computational cost, because the number of possible sets (baskets) grows exponentially with the number of items in the catalog. The issue is not trivial, even when we want to be able to add only one item to an existing set, and becomes even harder when we want to add more than one item with the intent of maximizing the diversity of the final recommended set.

Refs. [9, 10] address this combinatorial problem using a model based on determinantal point processes (DPPs) for basket completion. DPPs are elegant probabilistic models of repulsion from quantum physics, which are used for a variety of tasks in machine learning [17]. They allow sampling a diverse set of points, with similarity and popularity encoded using a positive semi-definite matrix called the kernel. Efficient algorithms for marginalization and conditioning DPPs are available. From a practical perspective, learning the DPP kernel is a challenge because the associated likelihood is non-convex, and learning it from observed sets of items is conjectured to be NP-hard [17].

For basket completion it is natural to consider that sets are the baskets which converted to sales. In this setting, the DPP is parameterized by a kernel matrix of size $p \times p$, where p is the size of the catalog. Thus the number of parameters to fit grows quadratically with p, and the computational complexity for learning, prediction, and sampling grows cubicly with p. As learning a full-rank DPP is hard, [10] proposes regularizing the DPP by constraining the kernel to be low rank. This regularization also improves generalization and offers more diversity in recommendations, without hurting predictive performance. In many settings the predictive quality is

KDD '19, August 4–8, 2019, Anchorage, AK, USA

ACM ISBN 978-1-4503-6201-6/19/08…$15.00
https://doi.org/10.1145/3292500.3330952

also improved, making the DPP a very desirable tool for modeling baskets. Moreover, the low-rank assumption also offers better runtime performance compared to a full-rank DPP.

Nevertheless, because of the definition of the DPP, as described in the *Model* section, this low-rank assumption for the kernel means that any possible baskets with more items than the chosen rank will receive a probability estimate of 0. This approach is thus impossible to use for large baskets, and some other regularizations of the DPP kernel may be more appropriate. Also, because of the symmetry property of the DPP kernel, it is impossible to model ordered correlations. However, the order in which items are added to a shopping basket can play an important role in the basket completion task.

Contributions. The contributions of this paper are fourfold:

- We modify the constraints over the kernel to support large baskets; that is, we prevent the model from returning a probability of 0 for sets larger than the rank of the kernel.
- We model the probability over all baskets by adding a logistic function on the determinant computed from the DPP kernel. We adapt the training procedure to handle this nonlinearity, and evaluate our model on several real-world basket datasets.
- By leveraging tensor factorization, we propose a new way to regularize the kernel among the collection of items in the catalog. This approach also leads to enhanced predictive quality.
- We show that this new model, which we call the tensorized DPP, allows us to capture ordered basket completion. That is, we can leverage the information regarding the order in which items are added to a basket to improve predictive quality.

Furthermore, we show that these ideas can be combined for further improvements to predictive quality, allowing our tensorized DPP model to outperform state-of-the-art models by a large margin.

We begin by discussing related work. Following this, we introduce our proposed model, and then proceed with an evaluation of its effectiveness on several real-world datasets.

2 RELATED WORK

In addition to the previously discussed work, DPPs have been used for natural language processing in order to discover diverse threads of documents [11], and to enhance diversity in recommender systems [6]. Unlike in our application where we learn the kernel, in these applications the kernel is constructed using previously obtained latent factors, for instance using tf-idf [11]. These latent factors are scaled by a relevance score learned in a conventional fashion. For example, these relevance scores may represent the predicted rating of a particular user, or the similarity between the text in a document and the user query. Ultimately, these applications involve sampling from the DPP specified by this kernel to find the set with the highest probability, called *Maximum At Posteriori* or MAP, where the kernel parameters trade off between relevance and diversity. However, efficiently finding the MAP for a DPP is challenging, and this has lead to work on different sampling techniques. Ref. [12] relies on MCMC sampling, while [6] proposes a greedy solution based on Cholesky decomposition. Instead of looking for the MAP, [13] involves looking for the *Maximizing Induced Cardinality* (MIC) set, which defines the set that contains the largest number of items with which the user will interact. Finally, [4] breaks the symmetry assumption of DPP by introducing signed DPPs, in order to incorporate not only repulsive but also attractive interactions between items.

Several methods have been proposed for learning the DPP kernel matrix. Ref. [12] uses an expectation-maximization (EM) algorithm to learn a non-parametric form of the DPP kernel matrix. Ref. [21] proposes a fixed-point algorithm called Picard iteration, which is much faster than EM, but still slower than [10]. Bayesian learning methods have also been proposed to learn the DPP kernel [1, 9].

Improving diversity in recommender systems has also been studied without the use of DPPs, including, among other work, [7, 25, 29]. For instance, [7] relies on random walk techniques to enhance diversity. In [25], the authors propose trading off between the relevance of the recommendation and diversity by introducing a coverage function to force the algorithm to produce recommendations that cover different centers of the interests of each user. Finally, the authors of [29] propose transforming the problem of recommending items to users into recommending users to items. They introduce a modification of nearest-neighbor methods, and a probabilistic model that allows isolation of the popularity bias and favors less popular items.

Regarding basket completion, associative classifiers have long been the state-of-the-art [2], despite requiring very heaving computational load for training, and manual tuning for key parameter choices such as lift and confidence thresholds. Later work focuses on the task of purchase prediction by adapting collaborative filtering methods. Ref. [23] proposes a solution based on nearest-neighbor models, while [18] relies on binary logistic regression to predict if a user will purchase a given item. More recently, DPPs [9, 10] may now be considered among the class of models belonging to the new state-of-the-art for basket completion, in light of their effectiveness both in terms of accuracy and training speed. Finally, classic collaborative filtering models tailored for positive and unlabelled data [14, 16] may be effectively used for basket completion.

3 MODEL

Determinantal point processes (DPPs) were originally used to model a distribution over particles that exhibit a repulsive effect [30]. Recently, interest in leveraging this repulsive behavior has led to DPPs receiving increased attention within the machine learning community. Mathematically, discrete DPPs are distributions over discrete sets of points, or in our case items, where the model assigns a probability to observing a given set of items. Let $\mathcal{I}$ denote a set of items, and L the kernel matrix associated with the DPP whose entries encode item popularity and the similarity between items. The probability of observing the set $\mathcal{I}$ is proportional to the determinant of the principal submatrix of L indexed by the items in $\mathcal{I}$: $\mathbb{P}(\mathcal{I}) \propto \det L_{\mathcal{I}}$ [1]. Thus, if p denotes the number of items in the item catalog, the DPP is a probability measure on 2^p (the power set, or set of all subsets of p), while it contains only p^2 parameters. The kernel L encodes item popularities and the similarities between items, where the diagonal entry L_{ii} represents the popularity of item i, and the off-diagonal entry $L_{ij} = L_{ji}$ represents the similarity

[1] To define a probability measure on the DPP, the normalization factor is $\det(L + I)$, because $\sum_{\mathcal{I}} \det L_{\mathcal{I}} = \det(L + I)$.

between items i and j. A determinant can be seen as a volume from a geometric viewpoint, and therefore more diverse sets will tend to have larger determinants. For example, the probability of selecting two items i and j together can be computed as

$$\mathbb{P}[\{i,j\}] \propto \begin{vmatrix} L_{ii} & L_{ij} \\ L_{ji} & L_{jj} \end{vmatrix} = L_{ii}L_{jj} - L_{ij}^2 \tag{1}$$

In equation 1 we can see that the more similar i and j are, the less likely they are to be sampled together. The definition of the entries L_{ij} will therefore determine the repulsive behavior of the kernel for the task. For instance, if similarity is defined using image descriptors, then images of differing appearance will be selected by a DPP. On the other hand, if the entries L_{ij} are learned using previously observed sets, such as e-commerce baskets [10], then co-purchased items i and j are likely to be sampled by the DPP, and thus the "similarity" L_{ij} will be low. Since co-purchased items are likely to have some diversity, DPPs are a natural choice for modeling baskets containing purchased items. In an application such as a search engine or in document summarization, the kernel may be defined using feature descriptors $\psi_i \in \mathbb{R}^D$ (i.e tf-idf of the text), and a relevance score $q_i \in \mathbb{R}^+$ of each item i, such that $L_{ij} = q_i \psi_i^T \psi_j q_j$, which favors relevant items (large q_i) and discourages lists composed of similar items.

3.1 Logistic DPP

Our objective is to find a set of items that are most likely to be purchased together. We formulate this as a classification problem, where the goal is to predict if a specific set of items will generate a conversion from the user, that is, all the items will be bought together, which we denote as $Y \in \{0, 1\}$. We model the class label Y as a Bernoulli random variable with parameter $\phi(\mathcal{I})$, where $\mathcal{I}$ is the set of items, and ϕ is a function that we will define below:

$$p(y|\mathcal{I}) = \phi(\mathcal{I})^y (1 - \phi(\mathcal{I}))^{1-y} \tag{2}$$

We model the function ϕ using a DPP.

We assume that there exists a latent space such that diverse items in this space are likely to be purchased together. Similarly to [10], we assume a low-rank constraint on the kernel matrix $L \in \mathbb{R}^{p \times p}$, which we factorize as follows:

$$L = VV^T + D^2 \tag{3}$$

where $V \in \mathbb{R}^{p \times r}$ is a latent matrix, where each row vector i encodes the r latent factors of item i. D is a diagonal matrix that, and together with $||V_i||$, represents the intrinsic quality or popularity of each item. The squared exponent on D insures that we always have a valid positive semi-definite kernel. We then define $\phi(\mathcal{I}) \propto \det(V_{\mathcal{I},:} V_{\mathcal{I},:}^T + D^2) \geq 0$. Note that without the diagonal term, the choice of r would restrict the cardinality of the observable set, because $|\mathcal{I}| > r$ would imply $\phi(\mathcal{I}) = 0$ when $D \equiv 0$. Using this term will ensure that the success probability of any set will be positive, but the cross-effects will be lower for sets of cardinality higher than r. We also see that items with similar latent vectors are less likely to be sampled than items with different latent vectors, since similar vectors will produce a parallelotope with a smaller volume. To normalize the probability and encourage separation between vectors we use a logistic function on Φ such that:

$$\phi(I) = \mathbb{P}(y = 1|\mathcal{I}) \doteq 1 - \exp(-w \det L_{\mathcal{I}}) \tag{4}$$

$$\doteq \sigma(w \det L_{\mathcal{I}}) \tag{5}$$

Usually the logistic function is of the form $1/(1 + \exp(-w \det L_{\mathcal{I}}))$. However, in our case the determinant is always positive, since L is positive semi-definite, which would result in $\mathbb{P}(y = 1|\mathcal{I})$ always greater than 0.5 with such a function. By construction, our formulation allows us to obtain a probability between 0 and 1. Finally, $w \in \mathbb{R}$ is a scaling parameter, to be chosen by cross-validation, that insures that the exponential does not explode, since the diagonal parameter will be approximately 1.

Learning. In order to learn the matrix V we assume the existence of historical data $\{\mathcal{I}_m, y_m\}_{1 \leq m \leq M}$, where $\mathcal{I}_m$ is a set of items, and y_m is a label set to 1 if the set has been purchased, and 0 otherwise. This training data allows us to learn the matrices V and D by maximizing the log-likelihood of the data. To do so, we first write the click probability for all y as

$$\mathbb{P}(y|\mathcal{I}) = \sigma(w \det L_{\mathcal{I}})^y (1 - \sigma(w \det L_{\mathcal{I}}))^{1-y} \tag{6}$$

The log-likelihood $f(V, D)$ can then be written as

$$\begin{aligned} f(V,D) &= \log \prod_{m=1}^{M} \mathbb{P}(y_m|\mathcal{I}_m) - \frac{\alpha_0}{2} \sum_{i=1}^{p} \alpha_i (||V_i||^2 + ||D_i||^2) \\ &= \sum_{m=1}^{M} \log \mathbb{P}(y_m|\mathcal{I}_m) - \frac{\alpha_0}{2} \sum_{i=1}^{p} \alpha_i (||V_i||^2 + ||D_i||^2) \end{aligned}$$

Following [10], α_i is an item regularization weight that is inversely proportional to item popularity. The matrices V and D are learned by maximizing the log-likelihood using stochastic gradient ascent. One step of gradient ascent requires the computation of the inverse and the determinant of a symmetric matrix ($L_{\mathcal{I}}$, where $\mathcal{I}$ is the considered item set for this gradient step) which can be done in $O(f^3)$ or $O(f^{2.373})$ using optimized CW-like algorithms, where f corresponds to the number of items in $\mathcal{I}$. The optimization algorithm used for learning is shown in Algorithm 1. Further details on the optimization algorithm and the gradient equations are available in the supplementary material.

3.2 Tensorized DPP

We now propose a modification to the previously introduced model that is better suited for the basket completion task. To do so we enhance the logistic DPP for the basket completion scenario, where we model the probability that the user will purchase a specified additional item based on the items already present in the user's shopping basket. We formulate this using a tensor, where the goal is to predict whether the user will purchase a given candidate target item based on the user's basket. Each slice of the tensor will correspond to a candidate target item. In this setting there are as many problems to solve as there are items in the catalog, p (minus the items already in the basket). Learning one kernel per item to recommend, where each item is independent from all other items, would be impossible in practice and suffer from sparsity issues. Each item is present in only a fraction of the baskets, and thus each kernel will only receive a small fraction of the data to be learned. However, all items are not completely independent from one another. Thus to solve the sparsity issue we utilize a low-rank tensor inspired by the

Algorithm 1 Optimization algorithm for Logistic DPP model.

Input: $\alpha_0 \in \mathbb{R}$, $\beta \in \mathbb{R}$ the momentum coefficient, $m \in \mathbb{N}$ the minibatch size, $\varepsilon \in \mathbb{R}$ the gradient step, $t = 0$ the iteration counter, $T = 0$, past data $\mathcal{D} = \{\mathcal{I}_m, y_m\}_{1 \leq m \leq M}$.
Initialization: Compute item popularity and output regularization weights α_i.
Set $D_0 \sim \mathcal{N}(1, 0.01)$ on the diagonal and $\tilde{D}_0 \equiv 0$ the gradient accumulation on D.
Set $V_0 \sim \mathcal{N}(0, 0.01)$ everywhere and $\tilde{V}_0 \equiv 0$ the gradient accumulation on V.
while not converged **do**
  if $m(t+1) > M(T+1)$ **then**
    Shuffle $\mathcal{D}$ and set $T = T + 1$
  end if
  Update $(\tilde{V}_{t+1}, \tilde{D}_{t+1}) = \beta(\tilde{V}_t, \tilde{D}_t) + (1-\beta)\varepsilon \nabla f(V_t + \beta\tilde{V}_t, D_t + \beta\tilde{D}_t)$
  Update $V_{t+1} = V_t + \tilde{V}_{t+1}$
  Update $D_{t+1} = D_t + \tilde{D}_{t+1}$
end while

RESCAL decomposition [24]. We use a cubic tensor $K \in \mathbb{R}^{p \times p \times p}$, where each slice τ (noted K_τ) of K is the candidate item (low-rank) kernel. By assuming that the tensor K is low-rank, we are able to implement sharing of learned parameters between each item, as shown in the following equation:

$$K_\tau = V R_\tau^2 V^T + D^2 \tag{7}$$

where $V \in \mathbb{R}^{p \times r}$ are the item latent factors that are common to all candidates items, and $R_\tau \in \mathbb{R}^{r \times r}$ is a candidate item specific matrix that models the interactions between the latent components of each candidate item. In order to balance the degrees of freedom between candidate items and items already in the basket, we further assume that R_τ is a diagonal matrix. Therefore, the diagonal vector of R_τ models the latent factors of each candidate item, and the latent factors of the item can be seen as the relevance of the product for each latent factor. As is the case for the matrix D, the squared exponent on R_τ ensures that we always have a valid kernel. See Figure 1 for an illustration of the factorization. The probability that the candidate item τ is relevant for the set of items $\mathcal{I}$ already in the basket is

$$\mathbb{P}(y_\tau = 1 | \mathcal{I}) = \sigma(w \det K_{\tau,\mathcal{I}}) = 1 - \exp(-w \det K_{\tau,\mathcal{I}}) \tag{8}$$

Therefore, the log-likelihood $g(V, D, R) \doteq g$ is

$$g = \sum_{m=1}^{M} \log \mathbb{P}(y_\tau | \mathcal{I}_m) - \frac{\alpha_0}{2} \sum_{i=1}^{p} \alpha_i (||V_i||^2 + ||D_i||^2 + ||R^i||^2)$$

where each observation m is associated with a candidate item, and $\mathcal{I}_m$ is the set of basket items associated with an observation. As previously described, the matrices V, D, and $(R_\tau)_{\tau \in \{1, \cdots, p\}}$ are learned by maximizing the log-likelihood using stochastic gradient ascent. Just as with the logistic DPP model, one step of gradient ascent requires the computation of the inverse and the determinant of a symmetric the matrix $L_\mathcal{I}$), resulting in a $O(f^{2.373})$ complexity (with f the number of items in $\mathcal{I}$). The algorithm used for learning is shown in Algorithm 2. Further details on the optimization

Algorithm 2 Optimization algorithm for Tensorized DPP model.

Input: $\alpha_0 \in \mathbb{R}$, $\beta \in \mathbb{R}$ the momentum coefficient, $m \in \mathbb{N}$ the minibatch size, $\varepsilon \in \mathbb{R}$ the gradient step, $t = 0$ the iteration counter, $T = 0$, past data $\mathcal{D} = \{\mathcal{I}_m, y_m\}_{1 \leq m \leq M}$.
Initialization: Compute item popularity and output regularization weights α_i.
Set $D_0 \sim \mathcal{N}(1, 0.01)$ on the diagonal and $\tilde{D}_0 \equiv 0$ the gradient accumulation on D.
Set $V_0 \sim \mathcal{N}(0, 0.01)$ everywhere and $\tilde{V}_0 \equiv 0$ the gradient accumulation on V.
Set $R_{\tau,0} \sim \mathcal{N}(1, 0.01)$ on the diagonal for each item and $\tilde{R}_{\tau,0} \equiv 0$ the gradient accumulation on R_τ.
while not converged **do**
  if $m(t+1) > M(T+1)$ **then**
    Shuffle $\mathcal{D}$ and set $T = T + 1$
  end if
  Update $\left(\tilde{V}_{t+1}, \tilde{D}_{t+1}, (\tilde{R}_{\tau,t+1})_\tau\right) = \beta\left(\tilde{V}_t, \tilde{D}_t, (\tilde{R}_{\tau,t})_\tau\right) + (1-\beta)\varepsilon \nabla g(V_t + \beta\tilde{V}_t, D_t + \beta\tilde{D}_t, (R_{\tau,t} + \beta\tilde{R}_{\tau,t})_\tau)$
  Update $V_{t+1} = V_t + \tilde{V}_{t+1}$
  Update $D_{t+1} = D_t + \tilde{D}_{t+1}$
  Update $R_{\tau,t+1} = R_{\tau,t} + \tilde{R}_{\tau,t+1}$ for all τ
end while

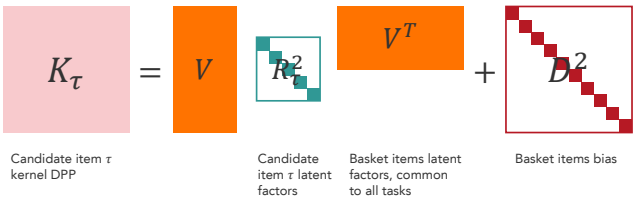

Figure 1: Illustration of the decomposition of the tensorized DPP.

algorithms and the gradient equations are available in the supplementary material.

Generalization to higher-order interactions. In basket completion applications, it may be interesting to try to recommend several items at the same time. This can be done using a greedy approach. That is, we first complete the basket with an initial product, consider this augmented basket (with the new product) as a new basket, and then complete it. A more direct approach, and perhaps more suitable for capturing higher-order interactions between items, would be to generalize Eq. 7. Here we propose a higher-order version of our model, and leave performance evaluation of this model for future work. Let d be the number of items to recommend and let $\tau = \{\tau_1, \cdots, \tau_d\} \in [p]^d$. We then define the kernel K_τ as:

$$K_\tau = V \prod_{k=1}^{d} R^2_{(d),\tau_d} V^T + D^2 \tag{9}$$

where each $R_{(d),\tau_d} \in \mathbb{R}^{r \times r}$ is a diagonal matrix.

3.3 Prediction

As discussed previously, sampling from a DPP can be a difficult problem, and various solutions have been proposed [6, 12]. Although

sampling the best set among all possible sets has been conjectured to be NP-hard, our goal is to find only the best item to complete the basket. In such applications a greedy approach can be applied effectively, particularly with the low-rank structure of our model. In addition, [10] proposed an effective method for the basket completion scenario that involves conditioning the DPP, which can be applied to our logistic DPP model.

4 EXPERIMENTS

We evaluate the performance of our model on the basket completion problem on several real-world datasets, and compare to several state-of-the-art baselines.

- **Our models**. To understand the impact of the different components of our model compared to the low-rank DPP model, we evaluated the following versions of our model:
  - **Logistic DPP**: This version of our model is similar to the low-rank DPP model, with the addition of the logistic function. To determine what item to recommend we use a greedy approach, where we select the next item such that the probability of the basket completed with this item is the largest. We used $w = 0.01$.
  - **Tensorized log-DPP without bias**: In this version of the model we set $D \equiv 0$, which allows us to measure the impact of capturing the item bias in a separate matrix. The matrix V encodes the latent factors of items present in the basket, while each matrix R_τ encodes the latent factors of each target item τ that can be added to a basket. $w = 0.01$.
  - **Tensorized log-DPP**: This is the full version of our model, with bias enabled. We used $w = 0.01$.

  Our datasets do not provide explicit negative information. To overcome this issue, similar to the approach to described in [22], we generate negative feedback for our models from observed baskets by sampling a random item among items not in the basket. This approach could be improved through better negative sampling strategies, but since this is not part of our primary contributions we leave this investigation for future work. Our results show that even this simple strategy achieves very good predictive performance.
- **Baselines**. The primary goal of our work is to improve state-of-the-art results provided by DPPs and introduce new modeling enhancements to DPPs. However, for the sake of completeness we also compare with other strong baseline collaborative filtering models.
  - **Poisson Factorization** (PF) [14] is a probabilistic matrix factorization model generally used for recommendation applications with implicit feedback. Since our datasets contain no user id information, we consider each basket to be a different user, and thus there are as many users as baskets in the training set. In practice this can cause issues with high memory consumption, since the number of baskets can be very large.
  - **Factorization Machines** (FMs) [26] is a general approach that models dth-order interactions using low-rank assumptions. FMs are usually used with $d = 2$, since this corresponds to classic matrix factorization, and because complexity increases linearly with d. Additionally, there is no open-source FM implementation that supports $d > 2$. For these reasons, we use $d = 2$ in our experiments. As with PF, to learn the FM model we consider each basket as a unique user. For fairness in comparison with our models, we also tried FM with negative sampling based on item popularity. However, we did not see any substantial improvement in model performance when using this negative sampling approach.
  - **Low-Rank DPP** [10] is a low-rank DPP model, suitable for basket completion, where the determinant of the submatrix of the kernel corresponds to the probability that all the items will be bought together in a basket.
  - **Bayesian Low-Rank DPP** [9] is the Bayesian version of the low-rank DPP model.
  - **Associative Classifier** (AC) is an algorithm that computes the support of a purchased set of items in order to obtain completion rules. As in [10], we used the Classification Based on Associations (CBA) algorithm [20], available at [8], with minimum support of 1.0% and maximum confidence thresholds of 20.0%. Unlike other models, AC does not provide estimates for all possible sets. Therefore, we cannot compute results for some metrics used in our evaluation, such as MPR (described below).
  - **Recurrent Neural Network** This RNN model [15] is adapted for session-based recommender systems. The RNN requires ordered sequences, and thus we only evaluate this model on the Instacart dataset (described below), where the order in which items were added to each basket is available. We use the implementation of this model available from [27].

For all models we cross-validated different hyperparameter settings, such as the number of latents factors (5, then from 10 to 60 in steps of 10, plus the size of the largest basket; for the Belgian dataset we rounded up to 75 in our models even if the largest basket is of size 76, since it does not reduce the performance of our models) and regularization strength (0.01, 0.1, and 1.0), and report the best results here. In the interest of reproducibility, all code used for our experiments is available at [31]

DATASETS. For our basket completion experiments we use the following four datasets. The first three datasets contain unordered baskets, while the last dataset contains ordered baskets:

- **Amazon Baby Registries** is a public dataset consisting of 110,006 registries and 15 disjoint registry categories. For the purposes of comparison with [9], we perform two experiments. The first experiment is conducted using the diaper category, which contains 100 products and approximately 10,000 baskets, composed of 2.4 items per basket on average. The second experiment is performed on the concatenation of the diaper, apparel, and feeding categories (sometimes denoted here as D.A.F, for Diaper+Apparel+Feedings), which contains 300 products and approximately 17,000 baskets, composed of 2.6 items per basket on average. The item categories are disjoint; for example, no basket containing diaper products will contain apparel products. This concatenation of disjoint categories can present difficulties for classic matrix

factorization models [9], which may prevent these models from learning a good embedding of items.

- **Belgian Retail Supermarket** is a public dataset [3] that contains 88,163 sets of items that have been purchased together, with a catalog of 16,470 unique items. Each basket contains 9.6 items on average. AC cannot be trained on this dataset because of scaling issues associated with large item catalogs.
- **UK retail dataset** is a public dataset [5] that contains 22,034 sets of items that have been purchased together, from a catalog of 4,070 unique items. This dataset contains transactions from a non-store online retail company that primarily sells unique all-occasion gifts, and many customers are wholesalers. Each basket contains 18.5 items on average, with a number of very large baskets. Modeling these large baskets requires using a very large number of latent factors for the low-rank DPP, leading to somewhat poor results for this model. This is not an issue for our model, due to the item bias that is captured in a separate matrix. However, for purposes of comparison, we removed all baskets containing more than 100 items from this dataset; note that the low-rank DPP still requires 100 latent factors to model these baskets. AC could not be trained on this dataset because it does not scale to large item catalogs.
- **Instacart** is, to the best of our knowledge, the only public dataset [2] that contains the order in which products were added to baskets. It is composed of three datasets containing online grocery shopping behavior for more than 200,000 Instacart users: a "train" dataset, a "test" dataset, and a "prior" dataset. To keep training time reasonable, we use only the "train" dataset in our experiments, and remove items that appear less than 15 times and baskets of size less than 3. This results in a dataset containing 700,052 sets of items and 10,531 unique items.

METRICS. To evaluate the performance of each model we compute the Mean Percentile Rank and precision@K for $K = 5, 10$, and 20:

- **Mean Percentile Rank (MPR)**: Given a basket B, we compute the percentile rank PR_{i_B} of the held-out item, i_B. Let $p_i \doteq P(Y = 1|B)$. Then

$$\mathrm{PR}_{i_B} = \frac{\sum_{i=1}^{p} \mathbb{I}(p_{i_B} \geq p_i)}{p} \times 100\% \tag{10}$$

  The MPR is the average PR over all baskets in the test set:

$$\mathrm{MPR} = \frac{\sum_{B \in \mathcal{T}} \mathrm{PR}_{i_B}}{|\mathcal{T}|} \tag{11}$$

  where $\mathcal{T}$ is the set of all baskets in the test set. A MPR of 100% means that the held-out item always receives the highest predictive score, while a MPR of 50% corresponds to a random sorting. Higher MPR scores are better.
- **Precision@K** is the fraction of test baskets where the held-out item is in the top K ranked items.

$$\text{precision@}K = \frac{\sum_{B \in \mathcal{T}} \mathbb{I}(\mathrm{rank}_{i_B} \leq K)}{|\mathcal{T}|} \tag{12}$$

  Higher precision@K scores are better.

[2] https://www.instacart.com/datasets/grocery-shopping-2017

We evaluated the predictive quality of our models for both unordered and ordered basket completion. Recall that for unordered baskets, there is no information regarding the order in which items are added to baskets, while ordered baskets do contain such ordering structure. We use the Amazon, Belgian retail, and UK retail datasets for our unordered basket experiments, while the Instacart dataset is used for our ordered basket experiments. For all experiments we use a random split of 70% of the data for training, and 30% for testing.

4.1 Results for Unordered Baskets

For unordered basket experiments, since there is no way to know the last item that was added to a basket, we remove one item at random from each basket in the test set. We then evaluate the model prediction according to the predicted score of this removed item using the metrics described below.

Looking at Table 1, we see that conventional collaborative filtering models sometimes have difficulty providing good recommendations in the basket-completion setting. Perhaps more surprising, but already described in [9], is that for the Amazon datasets, PF provides MPR performance that is approximately equivalent to a random model. For the Amazon diaper dataset this poor performance may be a result of the small size of each basket (around 2.4 items per basket on average), thus each "user" is in a cold start situation, and it is therefore difficult to provide good predictions. Poor performance on the diaper+apparel+feedings Amazon's dataset may result from the fact that, apart from the small basket size of 2.61 items on average, this dataset is composed of three disjoint categories. These disjoint categories can break the low-rank assumption for matrix factorization-based models, as discussed in [9]. This issue is somewhat mitigated in FM, due to the integration of an item bias into the model. This item bias allows the model to capture item popularity and thus provide acceptable performance in some cases.

Finally, the DPP-based models generally outperform the FM model. This is likely due to the fact that DPP models are able to capture higher-order interactions within baskets, while FM is only able to capture second-order interactions, since $d = 2$ for this model.

Low-Rank DPP vs. Tensorized DPP. We now turn to a performance comparison between our primary baseline, the low-rank DPP model, and our tensorized DPP model. From Table 1, we see that our approaches provide a substantial increase in performance for both Amazon datasets, with relative improvements of between 10% and 70%. One factor that accounts for this performance improvement is that unlike the low-rank DPP, which models the probability that a set of items will be bought together, our approach directly models the basket completion task. In our tensorized DPP model, the extra dimensions allow the model to capture the correlation between each item in the basket and the target item, as well as the global coherence of the set.

Regarding the three-category Amazon dataset, a good model should not be impacted by the fact the all of the three categories are disjoint. Therefore, the precision@K scores should be approximately the same for both the single-category and three-category datasets, since we observe similar performance for each category independently. Since the MPR is 78% for one category, the MPR on the three-category dataset should be approximately 93%, since on

model	dataset	r	MPR	Prec.@5	Prec.@10	Prec.@20
Associative Classifier★	Amazon (diaper)	-	-	16.66	16.66	16.66
Poisson Factorization★	Amazon (diaper)	40	50.30	4.78	10.03	19.90
Factorization Machines	Amazon (diaper)	5	67.92	24.01	32.62	46.25
Low Rank DPP★	Amazon (diaper)	30	71.65	25.48	35.80	49.98
Bayesian Low Rank DPP★	Amazon (diaper)	30	72.38	26.31	36.21	51.51
Logistic DPP	Amazon (diaper)	50	71.08	23.7	34.01	48.44
tensorized logDPP no bias	Amazon (diaper)	50	77.5	32.7	45.77	61.00
tensorized logDPP	Amazon (diaper)	50	**78.41**	**34.73**	**47.42**	**62.58**
Associative Classifier★	Amazon (D.A.F)	-	-	4.16	4.16	4.16
Poisson Factorization★	Amazon (D.A.F)	40	51.36	4.16	5.88	9.08
Factorization Machines	Amazon (D.A.F)	5	65.21	10.62	16.71	24.20
Low Rank DPP★	Amazon (D.A.F)	30	70.10	13.10	18.59	26.92
Bayesian Low Rank DPP★	Amazon (D.A.F)	30	70.55	13.59	19.51	27.83
Logistic DPP	Amazon (D.A.F)	60	69.61	12.65	19.8	27.86
tensorized logDPP no bias	Amazon (D.A.F)	60	88.77	18.33	28.00	43.57
tensorized logDPP	Amazon (D.A.F)	60	**89.80**	**20.53**	**30.86**	**45.79**
Poisson Factorization★	Belgian Retail Supermarket	40	87.02	21.46	23.06	23.90
Factorization Machines	Belgian Retail Supermarket	10	65.08	20.85	21.10	21.37
Low Rank DPP★	Belgian Retail Supermarket	76	88.52	21.48	23.29	25.19
Bayesian Low Rank DPP★	Belgian Retail Supermarket	76	89.08	21.43	23.10	25.12
Logistic DPP	Belgian Retail Supermarket	75	87.35	21.17	23.11	25.77
tensorized logDPP no bias	Belgian Retail Supermarket	75	87.42	21.02	23.35	25.13
tensorized logDPP	Belgian Retail Supermarket	75	87.72	21.46	23.37	25.57
Poisson Factorization	UK Retail	100	73.12	1.77	2.31	3.01
Factorization Machines	UK Retail	5	56.91	0.47	0.83	1.50
Low Rank DPP	UK Retail	100	**82.74**	3.07	4.75	7.60
Bayesian Low Rank DPP†	UK Retail	100	61.31	1.07	1.91	3.25
Logistic DPP	UK Retail	100	75.23	3.18	4.99	7.83
tensorized log DPP no bias	UK Retail	100	77.67	3.82	5.98	9.11
tensorized logDPP	UK Retail	100	78.25	**4.00**	**6.20**	**9.40**

Table 1: [Unordered Baskets] Results for all models on all datasets. r denotes the number of latent factors. The best results for each dataset are in bold. Models results marked with a ⋆ come directly from [9], where more baselines can be found. Other models have been retrained for this paper with the same training and testing set sizes as [9]. † Usually the Bayesian low-rank DPP shows little improvement over the standard low-rank DPP, however for the UK retail dataset we had to reduce the number of samples used for learning because of high memory consumption, which may explain the poor predictive performance for this experiment.

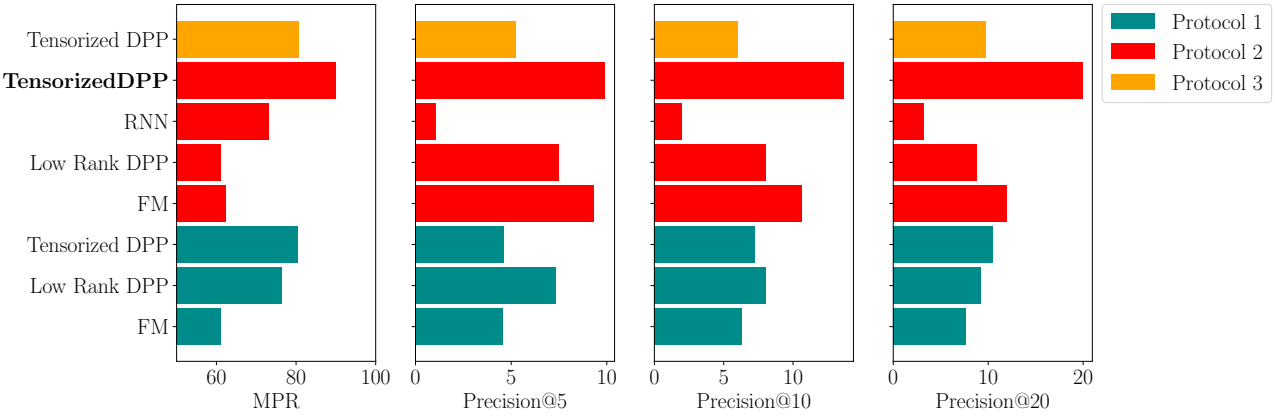

Figure 2: [Ordered Baskets] Performance of the models on Instacart dataset for the three protocols. Protocols 1, 2, and 3 correspond to predicting a randomly removed item, predicting the last added item with last-item removal in the training set (for the tensorized DPP), and predicting the last added item in the test set with random item removal in the training set, respectively. By cross-validation, all models used 80 latent factors, except FM, which used 5 latent factors. The best performance is achieved with our tensorized DPP model, particularly in protocol 2, when involves predicting the last added item; this protocol is the closest to a real-world application.

average for each category the target item is in the 22nd position over the 100 items in the single-category catalog. Therefore, the target item should be in the 22nd position over the 300 items in the three-category catalog, resulting in a MPR of $1 - 22/300 = 93\%$. Our models come close to these numbers, but still exhibit some small degradation for the three-category dataset. Finally, we note that for this dataset, a model that samples an item at random from the right category would have a precision@20 of 20%, since there are 100 items per category. The low-rank DPP model provides close to this level of performance. Taken together, these observations indicate our model is robust to the disjoint category problem, and explains the 70% relative improvement we see for our model on the precision@20 metric. On the UK Retail dataset the improvements of our algorithm are still substantial for precision@K, with a relative improvement of between 20% and 30% (MPR is down by 5%). We also observe the same decrease in MPR for our logistic DPP model, but precision@K is similar to the low-rank DPP. On the Belgian Retail dataset we see that all models provide similar performance. For this dataset, baskets come from an offline supermarket, where it is possible that customers commonly purchased similar products at specific frequencies. Consequently it may be easy to capture frequent associations between purchased items, but very difficult to discover more unusual associations, which may explain why all models provide approximately the same performance.

Logistic DPP vs. Tensorized DPP. To better understand the incremental performance of our model, we focus on the results of the logistic DPP and the tensorized log-DPP models. We see that the logistic model does not improve over the low-rank DPP on average, indicating that the logistic component of the model does not contribute to improved performance. However, we argue that this formulation may still be valuable in other classification applications, such as those with explicit negative feedback. For the tensorized log-DPP model, we see that the version of this model without bias is responsible for almost all of the performance improvement. Some additional lift is obtained when capturing the item popularity bias in a separate matrix. Since most of the gain comes from the tensorized kernel, one may ask if we could use the tensorized kernel without the logistic function and obtain similar results. We believe that this is not the case for two reasons. First, since we are clearly in a classification setting, it is more appropriate to use a logistic model that is directly tailored for such applications. Second, without the logistic function, each slice of the tensor should define a probability distribution, meaning that the probability of purchasing an additional product should sum to one over all possible baskets. However, we could add an arbitrarily bad product that would never be purchased, resulting in a probability of zero for buying that item in any basket, which would break the distributional assumption.

4.2 Results for Ordered Baskets

Recall that ordered baskets contain information regarding the order in which items are added to baskets, which may provide additional signal for the basket completion task. To evaluate the ability of our model to capture ordered basket completions, we performed three experimental protocols on the Instacart dataset, which contains ordered sequences of items added to baskets. Before introducing the different protocols, let us recall that the low-rank DPP is trained to capture item co-occurrence probabilities within sets, the FM model is trained to predict whether a set of items is going to be purchased together, and the tensorized DPP is trained to predict if a specified target item should be added to a given set of items. Each protocol varies in the way that we remove the item to predict from the basket:

(1) As with previous experiments, we remove one item at random from each basket. For the low-rank DPP and FM models this item removal is performed only for baskets in the test set. We do not remove items from baskets in the training set, since these baskets are used to learn inter-item correlation patterns that are applied to new baskets. For the tensorized DPP, we perform item removal for both the training and test sets. Item removal in the training set is appropriate for the tensorized DPP since the removed item corresponds to the target item for this model.
(2) We remove the last item added to each basket. For the low-rank DPP and FM models this is done only for the test set (since the training procedure does not include a target item), and for the tensorized DPP this done for both the training and test sets. Since we consider ordered sequences, we also evaluate the RNN model using this protocol, where item removal is done only for the test set. Since in practice the basket completion recommendation is performed at the end of the user session, just before the basket purchase, this protocol corresponds to a real-world application setting.
(3) We remove one item at random from each basket in the training set, and the last item added to each basket in the test set. Here we evaluate only the performance of the tensorized DPP model, since it is the only model that involves item removal in the training set. Since the low-rank DPP and RNN are trained on complete sets and sequences, respectively, removing one at random from each set or sequence during training will only result in reduced predictive performance.

If the order in which items are added into the basket contains no signal, then for any model, the performance of protocols (2) and (3) would be the same. However, looking at Table 2 and comparing the tensorized DPP results for protocols (2) and (3), we see that our model performs much better when predicting the last item added to a basket when training is also done by removing the last item added (protocol (3)), with MPR increasing from 80.65% in protocol (3) to 90.07% in protocol (2), and precision@20 increasing from 9.72 to 19.97. This allows us to conclude that the order in which items are added to the basket is important and can be captured by our model. Next, when comparing the results of protocols (1) and (2), we see that tensorized DPP performance is lower when the model is trained to predict a randomly removed item than when trained to predict the last item added (MPR of 80.46% and precision @20 of 10.51 for protocol (1)), while we see that this pattern is reversed for the low-rank DPP (MPR of 76.46% and precision @20 of 9.23 for protocol (1) vs MPR of 61.16% and precision @20 of 8.8 for protocol (2)). This indicates that the low-rank DPP, although well suited to modeling item co-occurrence probabilities, is unable to capture ordered basket completion. Finally, we see, surprisingly, that the RNN model does not provide good performance for this task. This relatively poor performance may come from the fact that the basket

lengths are too small in this dataset for the RNN to learn the item sequences within baskets correctly.

5 CONCLUSION AND FUTURE WORK

In this paper we have proposed an extension of the DPP model that leverages ideas from multi-class classification and tensor factorization. While our model can be applied to a number of machine learning problems, we focus on the problem of basket completion. We have shown through experiments on several datasets that our model provides significant improvements in predictive quality compared to a number of competing state-of-the-art approaches, and can appropriately capture ordered basket completion. In future work we plan to test the performance of our proposed higher-order method and to investigate other applications of our model, such as user conversion prediction, attribution, and adversarial settings in games. We also plan to investigate better negative sampling methods for positive-only and unlabelled data. Finally, we also plan to investigate other types of loss functions, such as hinge loss, and other types of link functions for DPPs, such as the Poisson function, to tailor DPPs for regression problems. We believe that this work will allow us to customize DPPs so that they are suitable for other applications involving complex combinatorial structure.

REFERENCES

[1] Raja Hafiz Affandi, Emily B. Fox, Ryan P. Adams, and Benjamin Taskar. 2014. Learning the Parameters of Determinantal Point Process Kernels. In Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014. 1224–1232. http://jmlr.org/proceedings/papers/v32/affandi14.html

[2] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. 1993. Mining Association Rules Between Sets of Items in Large Databases. SIGMOD Rec. 22, 2 (June 1993), 207–216. https://doi.org/10.1145/170036.170072

[3] Tom Brijs, Gilbert Swinnen, Koen Vanhoof, and Geert Wets. 1999. Using Association Rules for Product Assortment Decisions: A Case Study. In Knowledge Discovery and Data Mining. 254–260.

[4] Victor-Emmanuel Brunel. 2018. Learning Signed Determinantal Point Processes through the Principal Minor Assignment Problem. In Advances in Neural Information Processing Systems 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 7376–7385. http://papers.nips.cc/paper/7966-learning-signed-determinantal-point-processes-through-the-principal-minor-assignment-problem.pdf

[5] Daqing Chen, Sai Laing Sain, and Kun Guo. 2012. Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining. Journal of Database Marketing & Customer Strategy Management 19, 3 (01 Sep 2012), 197–208. https://doi.org/10.1057/dbm.2012.17

[6] Laming Chen, Guoxin Zhang, and Eric Zhou. 2018. Fast Greedy MAP Inference for Determinantal Point Process to Improve Recommendation Diversity. In Advances in Neural Information Processing Systems 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 5627–5638. http://papers.nips.cc/paper/7805-fast-greedy-map-inference-for-determinantal-point-process-to-improve-recommendation-diversity.pdf

[7] Fabian Christoffel, Bibek Paudel, Chris Newell, and Abraham Bernstein. 2015. Blockbusters and Wallflowers: Accurate, Diverse, and Scalable Recommendations with Random Walks. In Proceedings of the 9th ACM Conference on Recommender Systems (RecSys '15). ACM, New York, NY, USA, 163–170. https://doi.org/10.1145/2792838.2800180

[8] F. Coenen. 2005. TLUCS KDD implementation of CBA (classification based on associations).

[9] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. 2016. Bayesian Low-Rank Determinantal Point Processes. In Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016. 349–356. https://doi.org/10.1145/2959100.2959178

[10] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. 2017. Low-Rank Factorization of Determinantal Point Processes. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA. 1912–1918. http://aaai.org/ocs/index.php/AAAI/AAAI17/paper/view/14657

[11] Jennifer Gillenwater, Alex Kulesza, and Ben Taskar. 2012. Discovering Diverse and Salient Threads in Document Collections. In Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL '12). Association for Computational Linguistics, Stroudsburg, PA, USA, 710–720. http://dl.acm.org/citation.cfm?id=2390948.2391026

[12] Jennifer A Gillenwater, Alex Kulesza, Emily Fox, and Ben Taskar. 2014. Expectation-Maximization for Learning Determinantal Point Processes. In Advances in Neural Information Processing Systems 27, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3149–3157.

[13] Jennifer A Gillenwater, Alex Kulesza, Sergei Vassilvitskii, and Zelda E. Mariet. 2018. Maximizing Induced Cardinality Under a Determinantal Point Process. In Advances in Neural Information Processing Systems 31, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.). Curran Associates, Inc., 6911–6920. http://papers.nips.cc/paper/7923-maximizing-induced-cardinality-under-a-determinantal-point-process.pdf

[14] Prem Gopalan, Jake M. Hofman, and David M. Blei. 2013. Scalable Recommendation with Poisson Factorization. abs/1311.1704 (2013). arXiv:1311.1704

[15] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. CoRR abs/1511.06939 (2015). arXiv:1511.06939 http://arxiv.org/abs/1511.06939

[16] Yifan Hu, Yehuda Koren, and Chris Volinsky. 2008. Collaborative Filtering for Implicit Feedback Datasets. In Proceedings of the 2008 Eighth IEEE International Conference on Data Mining (ICDM '08). IEEE Computer Society, Washington, DC, USA, 263–272. https://doi.org/10.1109/ICDM.2008.22

[17] Alex Kulesza and Ben Taskar. 2012. Determinantal Point Processes for Machine Learning. Now Publishers Inc., Hanover, MA, USA.

[18] Jong-Seok Lee, Chi-Hyuck Jun, Jaewook Lee, and Sooyoung Kim. 2005. Classification-based Collaborative Filtering Using Market Basket Data. Expert Syst. Appl. 29, 3 (Oct. 2005), 700–704. https://doi.org/10.1016/j.eswa.2005.04.037

[19] G. Linden, B. Smith, and J. York. 2003. Amazon.com recommendations: item-to-item collaborative filtering. IEEE Internet Computing 7, 1 (Jan 2003), 76–80. https://doi.org/10.1109/MIC.2003.1167344

[20] Bing Liu, Wynne Hsu, and Yiming Ma. 1998. Integrating Classification and Association Rule Mining. In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining (KDD'98). AAAI Press, 80–86. http://dl.acm.org/citation.cfm?id=3000292.3000305

[21] Zelda Mariet and Suvrit Sra. 2015. Fixed-point algorithms for determinantal point processes. NIPS (2015). http://arxiv.org/abs/1508.00792

[22] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In Advances in Neural Information Processing Systems 26, C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger (Eds.). Curran Associates, Inc., 3111–3119. http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

[23] Andreas Mild and Thomas Reutterer. 2003. An improved collaborative filtering approach for predicting cross-category purchases based on binary market basket data. Journal of Retailing and Consumer Services (2003).

[24] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A Three-Way Model for Collective Learning on Multi-Relational Data. In Proceedings of the 28th International Conference on Machine Learning (ICML-11) (ICML '11), Lise Getoor and Tobiasa Scheffer (Eds.). ACM, New York, NY, USA, 809–816.

[25] Shameem A. Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. 2016. A Coverage-Based Approach to Recommendation Diversity On Similarity Graph. In Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16). ACM, New York, NY, USA, 15–22. https://doi.org/10.1145/2959100.2959149

[26] Steffen Rendle. 2010. Factorization Machines. In Proceedings of the 2010 IEEE International Conference on Data Mining (ICDM '10). IEEE Computer Society, Washington, DC, USA, 995–1000. https://doi.org/10.1109/ICDM.2010.127

[27] Songweiping. 2017. Github. https://github.com/Songweiping/GRU4Rec_TensorFlow.

[28] Choon Hui Teo, Houssam Nassif, Daniel Hill, Sriram Srinivasan, Mitchell Goodman, Vijai Mohan, and S.V.N. Vishwanathan. 2016. Adaptive, Personalized Diversity for Visual Discovery. In Proceedings of the 10th ACM Conference on Recommender Systems (RecSys '16). ACM, New York, NY, USA, 35–38. https://doi.org/10.1145/2959100.2959171

[29] Saúl Vargas and Pablo Castells. 2014. Improving Sales Diversity by Recommending Users to Items. In Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14). ACM, New York, NY, USA, 145–152. https://doi.org/10.1145/2645710.2645744

[30] Anatoly M. Vershik and Yuri Yakubovichn. 2001. Asymptotic Combinatorics with Applications to Mathematical Physics. Springer-Verlag Berlin Heidelberg.

[31] Romain Warlop. 2018. Github. https://github.com/RomainWarlop/logisticDPP.

A SUPPLEMENTARY MATERIAL

A.1 Logistic DPP

Recall that the logistic DPP log-likelihood is:

$$\begin{aligned} f(V,D) &= \log \prod_{m=1}^{M} \mathbb{P}(y_m|\mathcal{I}_m) - \frac{\alpha_0}{2}\sum_{i=1}^{p}\alpha_i(||V_i||^2 + ||D_i||^2) \\ &= \sum_{m=1}^{M} \log \mathbb{P}(y_m|\mathcal{I}_m) - \frac{\alpha_0}{2}\sum_{i=1}^{p}\alpha_i(||V_i||^2 + ||D_i||^2) \end{aligned}$$

Optimization. We maximize the log-likelihood using stochastic gradient ascent with Nesterov's Accelerated Gradient, which is a form of momentum. To simplify notation, we define $[m] \doteq \mathcal{I}_m$ and $\sigma_m = \sigma(w \det L_{[m]})$. Let $i \in \{1, \cdots, p\}, k \in \{1, \cdots, r\}$.

lemma When D is fixed, the gradient of (13) with respect to V_{ik} is

$$\begin{aligned} \frac{\partial f}{\partial V_{ik}} &= 2w \sum_{m,\, i\in[m]} ([L_{[m]}^{-1}]_{:,i} V_{:,k}) \frac{y_m - \sigma_m}{\sigma_m} \det L_{\mathcal{I}_m} \\ &\quad -\alpha_0 \alpha_i V_{ik} \end{aligned} \tag{13}$$

Proof Without the regularization term we have

$$\frac{\partial f}{\partial V_{ik}} = \sum_{m,\, i\in[m]} \frac{y_m}{\sigma_m}\frac{\partial \sigma_m}{\partial V_{ik}} + \frac{1-y_m}{1-\sigma_m}\left(-\frac{\partial \sigma_m}{\partial V_{ik}}\right) \tag{14}$$

$$= w \sum_{m,\, i\in[m]} \frac{y_m - \sigma_m}{\sigma_m} \frac{\partial \det L_{[m]}}{\partial V_{ik}} \tag{15}$$

$$= w \sum_{m,\, i\in[m]} \frac{y_m - \sigma_m}{\sigma_m} \operatorname{tr}\left(L_{[m]}^{-1}\frac{\partial L_{[m]}}{\partial V_{ik}}\right) \det L_{\mathcal{I}_m} \tag{16}$$

$$= 2w \sum_{m,\, i\in[m]} ([L_{[m]}^{-1}]_{:,i} V_{:,k}) \frac{y_m - \sigma_m}{\sigma_m} \det L_{\mathcal{I}_m} \tag{17}$$

where (17) follows from

$$\left[\frac{\partial L_{[m]}}{\partial V_{ik}}\right]_{s,t} = V_{sk}\delta_{i,t} + V_{tk}\delta_{i,s} \tag{18}$$

Therefore,

$$\operatorname{tr}\left(L_{[m]}^{-1}\frac{\partial L_{[m]}}{\partial V_{ik}}\right) = \sum_{s,t}(V_{sk}\delta_{i,t} + V_{tk}\delta_{i,s})[L_{[m]}^{-1}]_{s,t} \tag{19}$$

$$\begin{aligned} &= \sum_{s}[L_{[m]}^{-1}]_{s,i}V_{sk} + \sum_{t}[L_{[m]}^{-1}]_{i,t}V_{tk} \\ &= 2\sum_{s}[L_{[m]}^{-1}]_{s,i}V_{sk} \end{aligned} \tag{20}$$

adding the derivative of the regularization term concludes the proof. □

lemma When V is fixed, the gradient of (13) with respect to D_i is

$$\begin{aligned} \frac{\partial f}{\partial D_{ii}} &= 2w \sum_{m,\, i\in[m]} ([L_{[m]}^{-1}]_{i,i} D_{i,i}) \frac{y_m - \sigma_m}{\sigma_m} \det L_{[m]} \\ &\quad -\alpha_0 \alpha_i D_{ii} \end{aligned} \tag{21}$$

Proof As shown previously, and without the regularization term, we have

$$\frac{\partial f}{\partial V_{ik}} = w \sum_{m,\, i\in[m]} \frac{y_m - \sigma_m}{\sigma_m} \operatorname{tr}\left(L_{[m]}^{-1}\frac{\partial L_{[m]}}{\partial D_{i,i}}\right) \det L_{\mathcal{I}_m} \tag{22}$$

Since,

$$\left[\frac{\partial L_{[m]}}{\partial D_{i,i}}\right]_{s,t} = 2D_{i,i}\delta_{s,i}\delta_{t,i}$$

$$\operatorname{tr}\left(L_{[m]}^{-1}\frac{\partial L_{[m]}}{\partial D_{i,i}}\right) = 2[L_{[m]}^{-1}]_{i,i}D_{i,i}$$

adding the derivative of the regularization term concludes the proof. □

A.2 Tensorized DPP

Recall that the tensorized DPP log-likelihood is:

$$g = \sum_{m}^{M} \log \mathbb{P}(y_\tau|[m]) - \frac{\alpha_0}{2}\sum_{i=1}^{p}\alpha_i(||V_i||^2 + ||D_i||^2 + ||R^i||^2)$$

Optimization. Since each observation m is attached to a candidate item, we denote τ_m as the tensor slice that corresponds to observation m. Thus we have $\sigma_m = \sigma(\det K_{\tau_m,[m]})$. When there is no ambiguity, we also denote $K_{[m]} \doteq K_{\tau_m,[m]}$. Let $i \in \{1, \cdots, p\}, k \in \{1, \cdots, r\}$.

lemma When D and R are fixed, the gradient of (23) with respect to V_{ik} is

$$\begin{aligned} \frac{\partial g}{\partial V_{ik}} &= 2w \sum_{m,\, i\in[m]} \frac{y_{\tau_m} - \sigma_m}{\sigma_m} R^2_{\tau_m,k,k} [K_{\tau_m,[m]}^{-1}]_{:,i} \\ &\quad \cdot V_{:,k} \det K_{\tau_m,[m]} - \alpha_0\alpha_i V_{ik} \end{aligned} \tag{23}$$

Proof Without the regularization term we have

$$\frac{\partial g}{\partial V_{ik}} = \sum_{m,\, i\in[m]} \frac{y_{\tau_m}}{\sigma_m}\frac{\partial \sigma_m}{\partial V_{ik}} + \frac{1-y_{\tau_m}}{1-\sigma_m}\left(-\frac{\partial \sigma_m}{\partial V_{ik}}\right) \tag{24}$$

$$= w \sum_{m,\, i\in[m]} \frac{y_{\tau_m} - \sigma_m}{\sigma_m} \frac{\partial \det K_{[m]}}{\partial V_{ik}} \tag{25}$$

$$= w \sum_{m,\, i\in[m]} \frac{y_m - \sigma_m}{\sigma_m} \operatorname{tr}\left(K_{[m]}^{-1}\frac{\partial K_{[m]}}{\partial V_{ik}}\right) \tag{26}$$

$$\cdot \det K_{[m]} \tag{27}$$

Since,

$$[K_\tau]_{s,t} - D^2_{s,t} = \sum_{j=1}^{r}[VR^2_\tau]_{s,j}V_{t,j} = \sum_{j=1}^{r} V_{s,j}R^2_{\tau,j,j}V_{t,j}$$

$$\left[\frac{\partial K_{[m]}}{\partial V_{ik}}\right]_{s,t} = R^2_{\tau,k,k}(V_{t,k}\delta_{s,i} + V_{s,k}\delta_{t,i})$$

Thus,

$$\operatorname{tr}\left(K_{[m]}^{-1}\frac{\partial K_{[m]}}{\partial V_{ik}}\right) = 2R^2_{\tau,k,k}\sum_{s\in[m]}[K_{\tau_m,[m]}^{-1}]_{s,i}V_{s,k}$$

adding the regularization term concludes the proof. □

lemma When V and R are fixed, the gradient of (23) with respect to $D_{i,i}$ is

$$\begin{aligned} \frac{\partial g}{\partial D_{i,i}} &= 2w \sum_{m,\, i\in[m]} \frac{y_{\tau_m} - \sigma_m}{\sigma_m} [K_{t_m,[m]}^{-1}]_{i,i} D_{i,i} \det K_{\mathcal{I}_m} \\ &\quad -\alpha_0\alpha_i D_{ii} \end{aligned} \tag{28}$$

Proof Similarly, without the regularization term, we have

$$\frac{\partial g}{\partial D_{i,i}} = w \sum_{m,\, i\in[m]} \frac{y_m - \sigma_m}{\sigma_m} \operatorname{tr}\left(K_{[m]}^{-1}\frac{\partial K_{[m]}}{\partial D_{i,i}}\right) \det K_{[m]} \tag{29}$$

Algorithm 3 Full optimization algorithm for tensorized DPP.

Input: $\alpha_0 \in \mathbb{R}$, $\beta \in \mathbb{R}$ the momentum coefficient, $m \in \mathbb{N}$ the minibatch size, $\varepsilon \in \mathbb{R}$ the gradient step, $t = 0$ the iteration counter, $T = 0$, past data $\mathcal{D} = \{\mathcal{I}_m, y_m\}_{1 \le m \le M}$.
Initialization: Compute item popularity and output regularization weights α_i.
Set $D_0 \sim \mathcal{N}(1, 0.01)$ on the diagonal and $\tilde{D}_0 \equiv 0$ the gradient accumulation on D.
Set $V_0 \sim \mathcal{N}(0, 0.01)$ everywhere and $\tilde{V}_0 \equiv 0$ the gradient accumulation on V.
if multitask **then**
  Set $R_{\tau,0} \sim \mathcal{N}(1, 0.01)$ on the diagonal for each item and $\tilde{R}_{\tau,0} \equiv 0$ the gradient accumulation on R_τ.
end if
while not converged **do**
  if $m(t+1) > M(T+1)$ **then**
    Shuffle $\mathcal{D}$ and set $T = T + 1$
  end if
  if multitask **then**
    Update $\left(\tilde{V}_{t+1}, \tilde{D}_{t+1}, (\tilde{R}_{\tau,t+1})_\tau\right) = \beta\left(\tilde{V}_t, \tilde{D}_t, (\tilde{R}_{\tau,t})_\tau\right) + (1-\beta)\varepsilon \nabla g(V_t + \beta\tilde{V}_t, D_t + \beta\tilde{D}_t, (R_{\tau,t} + \beta\tilde{R}_{\tau,t})_\tau)$ according to formulas (23), (28) and (32)
  else
    Update $(\tilde{V}_{t+1}, \tilde{D}_{t+1}) = \beta(\tilde{V}_t, \tilde{D}_t) + (1-\beta)\varepsilon \nabla f(V_t + \beta\tilde{V}_t, D_t + \beta\tilde{D}_t)$ according to formulas (13) and (21)
  end if
  Update $V_{t+1} = V_t + \tilde{V}_{t+1}$
  Update $D_{t+1} = D_t + \tilde{D}_{t+1}$
  if multitask **then**
    Update $R_{\tau,t+1} = R_{\tau,t} + \tilde{R}_{\tau,t+1}$ for all τ
  end if
end while

Using (28)

$$\left[\frac{\partial K_{[m]}}{\partial D_{i,i}}\right]_{s,t} = 2D_{i,i}\delta_{s,i}\delta_{t,i} \tag{30}$$

thus

$$\operatorname{tr}\left(K_{[m]}^{-1}\frac{\partial K_{[m]}}{\partial D_{i,i}}\right) = 2[K_{t_m,[m]}^{-1}]_{i,i}D_{i,i} \tag{31}$$

adding the regularization term concludes the proof. □

lemma When V and D are fixed, the gradient of (23) with respect to $R_{k,k}$ is

$$\frac{\partial g}{\partial R_{\tau,k,k}} = 2w \sum_{m,\tau \in [m]} \frac{y_\tau - \sigma_m}{\sigma_m} R_{\tau,k,k} K^{-1}{}_{[m]} \cdot V_{:,k} \cdot V_{:,k}^T \det K_{[m]} - \alpha_0\alpha_\tau R_{\tau,k,k}, \tag{32}$$

Proof Similarly, without the regularization term, we have

$$\frac{\partial g}{\partial R_{\tau,k,k}} = w \sum_{m,i \in [m]} \frac{y_\tau - \sigma_m}{\sigma_m} \operatorname{tr}\left(K_{[m]}^{-1}\frac{\partial K_{[m]}}{\partial R_{\tau,k,k}}\right) \det K_{[m]} \tag{33}$$

$$\tag{34}$$

Using (28)

$$\left[\frac{\partial K_{[m]}}{\partial R_{\tau,k,k}}\right]_{s,t} = 2R_{\tau,k,k}V_{s,k}V_{t,k} \tag{35}$$

we obtain

$$\operatorname{tr}\left(K_{[m]}^{-1}\frac{\partial K_{[m]}}{\partial R_{\tau,k,k}}\right) = 2R_{\tau,k,k}\sum_{s,t \in [m]} [K_{[m]}^{-1}]_{s,t}V_{s,k} \cdot V_{t,k} \tag{36}$$

$$\tag{37}$$

adding the regularization term concludes the proof. □