Learning Determinantal Point Processes by Sampling Inferred Negatives

Zelda Mariet* zelda@csail.mit.edu
Mike Gartrell† m.gartrell@criteo.com
Suvrit Sra* suvrit@mit.edu
∗ *Massachusetts Institute of Technology*
† *Criteo Research*

Abstract

Determinantal Point Processes (DPPs) have attracted significant interest from the machine-learning community due to their ability to elegantly and tractably model the delicate balance between quality and diversity of sets. We consider learning DPPs from data, a key task for DPPs; for this task, we introduce a novel optimization problem, *Contrastive Estimation (CE)*, which encodes information about "negative" samples into the basic learning model. CE is grounded in the successful use of negative information in machine-vision and language modeling. Depending on the chosen negative distribution (which may be static or evolve during optimization), CE assumes two different forms, which we analyze theoretically and experimentally. We evaluate our new model on real-world datasets; on a challenging dataset, CE learning delivers a considerable improvement in predictive performance over a DPP learned without using contrastive information.

1 Introduction

Careful selection of items from a large collection underlies many machine learning applications. Notable examples include recommender systems, information retrieval and automatic summarization methods, among others. Typically, the selected set of items must fulfill a variety of application specific requirements—e.g., when recommending items to a user, the *quality* of each selected item is important. This quality must be, however, balanced by the *diversity* of the selected items to avoid redundancy within recommendations.

But balancing quality with diversity is challenging: as the collection size grows, the number of its subsets grows exponentially. A model that offers an elegant, tractable way to achieve this balance is a Determinantal Point Process (DPP). Concretely, a DPP models a distribution over subsets of a ground set $\mathcal{Y}$ that is parametrized by a semi-definite matrix $\boldsymbol{L} \in \mathbb{R}^{|\mathcal{Y}|\times|\mathcal{Y}|}$, such that for any subset $A \subseteq \mathcal{Y}$,

$$\Pr(A) \propto \det(\boldsymbol{L}_A), \tag{1}$$

where $\boldsymbol{L}_A = [\boldsymbol{L}_{ij}]_{i,j\in A}$ is the submatrix of $\boldsymbol{L}$ indexed by A. Informally, $\det(\boldsymbol{L}_A)$ represents the volume associated with subset A, the diagonal entry L_{ii} represents the importance of item i, while entry $L_{ij} = L_{ji}$ encodes similarity between items i and j. Since the normalization constant of (1) is simply $\sum_{A\subseteq\mathcal{Y}} \det(\boldsymbol{L}_A) = \det(\boldsymbol{L}+\boldsymbol{I})$, we have $\Pr(A) = \det(\boldsymbol{L}_A)/\det(\boldsymbol{L}+\boldsymbol{I})$, which suggests why DPPs may be tractable despite their exponentially large sample space.

The key object defining a DPP is its kernel matrix $\boldsymbol{L}$. This matrix may be fixed *a priori* using domain knowledge [7], or as is more common in machine learning applications, learned from observations using maximum likelihood estimation (MLE) [20, 32]. However, while fitting observed subsets well, MLE for DPPs may also assign high likelihoods to unobserved subsets far from the underlying generative distribution [12]. MLE-based DPP models may thus have modes corresponding to subsets that are close in likelihood, yet differ in how close they are to the true data distribution. Such confusable modes reduce the quality of the learned model, hurting predictions.

Such concerns when learning generative models over huge sample spaces are not limited to the area of subset-selection: applications in image and text generation have been the driving force in developing techniques for generating high-quality samples. Among their innovations, a particularly successful technique uses generated samples as "negative samples" to train a discriminator, which in turn encourages generation of more realistic samples; this is the key idea behind the Generative

Adversarial Nets (GANs) introduced in [21]. These observations motivate us to investigate the use of DPP-generated samples as *negatives*, which we then incorporate into the DPP learning task to improve the modeling power of DPPs. Intuitively, negative samples are those subsets that are far from the true data distribution, but to which the DPP erroneously assigns high probability. As there is no closed form way to generate such idealized negatives, we approximate them via an external "negative distribution". More precisely, we introduce a novel DPP learning problem that incorporates samples from a negative distribution into traditional MLE. While the focus of our work is on generating the negative distribution *jointly* with $\boldsymbol{L}$, we also investigate outside sources of negative information. Ultimately, our formulation leads to an optimization problem harder than the original DPP learning problem; we show that even approximate solutions greatly improve the performance of the DPP model when evaluated on concrete tasks, such as identifying the best item to add to a subset of chosen objects (*basket-completion*).

Contributions. To our knowledge, this work is the first theoretical or empirical investigation of augmenting the DPP learning problem with negative information.

– Our first main contribution is the Contrastive Estimation (CE) model, which incorporates negative information through *inferred negatives* into the learning task.

– Next we introduce static and dynamic models for CE and discuss the theoretical and practical trade-offs of such choices. Static models leverage information that does not evolve over time, whereas dynamic models draw samples from a negative distribution that depends on the current model's parameters; dynamic CE posits an optimization problem worthy of independent study.

– Finally, we show how to learn CE models efficiently; furthermore, we show that the complexity of conditioning a DPP on a chosen sample can be brought down from $\mathcal{O}(|\mathcal{Y}|^2)$ to essentially $\mathcal{O}(|\mathcal{Y}|)$. This speedup helps dynamic CE while removing a major bottleneck in computing next-item predictions for a set.

Using findings obtained from extensive experiments conducted on small datasets, we show on a large dataset that CE learning significantly improves the modeling power of DPPs, as measured by metrics that evaluate a subset-selection model's performance on basket completion tasks.

2 Background and related work

First introduced to model fermion behavior by Macchi [31], DPPs have gained popularity due to their elegant balancing of quality and subset diversity. DPPs are studied both for their theoretical properties [27, 7, 1, 26, 19, 14, 28] and their machine learning applications: object retrieval [1], summarization [30, 12], sensor placement [25], recommender systems [17], neural network compression [33], and minibatch selection [45].

Gillenwater et al. [20] study DPP kernel learning via EM, while Mariet & Sra [32] present a fixed-point method. DPP kernel learning has leveraged Kronecker [34] and low-rank [16, 18] structures. Learning guarantees using DPP graph properties are studied in [44].

Aside from Tschiatschek et al. [43], Djolonga et al. [15], who learn a Facility LocatIon Diversity (FLID) distribution (as well as more complex FLIC and FLDC models) by contrasting it with a "negative" product distribution, little attention has been given to using negative samples to learn richer subset-selection models.

Nonetheless, leveraging negative information is a widely used in other applications. In object detection, negative mining corrects for the skewed simple-to-difficult negative distribution by training the model on its false positives [42, 11, 39]. In language modeling, Noise Contrastive Estimation (NCE) [23], which tasks the model with distinguishing positive samples from generated negatives, was first applied in [36] and has been instrumental in Word2Vec [35]. Since then, variants using adaptive noise [13] have been introduced. NCE is also the method used by Tschiatschek et al. [43] for subset-selection.

An alternate approach to negative samples within submodular language models was introduced as Contrastive Estimation in Smith & Eisner [40, 41]. Negative sampling is also used in Generative Adversarial Networks [21], where a generator network is trained by competing with an adversarial discriminative network which distinguishes between positives and generated negatives.

3 Learning DPPs with negative samples

Motivated by the similarities between DPP learning and crucial problems of structured prediction in other machine-learning fields, we introduce an optimization problem that leverages negative information. We refer to this problem as Contrastive Estimation (CE), due to its ties to a notion of the same name discussed by Smith & Eisner [40].

3.1 Contrastive Estimation

In conventional DPP learning, we seek to maximize determinantal volumes of sets drawn from the true distribution μ (that we wish to model), by solving the following MLE problem, where samples in the training set $\mathcal{A}^+$ are assumed to be drawn i.i.d.:

$$\text{Find } \boldsymbol{L} \in \operatorname*{argmax}_{\boldsymbol{L}\succeq 0} \phi_{\text{MLE}}(\boldsymbol{L}) \tag{2}$$
$$\phi_{\text{MLE}}(L) \triangleq \frac{1}{|\mathcal{A}^+|} \sum_{A\in\mathcal{A}^+} \log\det(\boldsymbol{L}_A) - \log\det(\boldsymbol{L}+\boldsymbol{I}).$$

We augment problem (2) to incorporate additional information from a *negative* distribution ν, which we wish to have the DPP distribution move away from. The ensuing optimization problem is the main focus of our paper.

Definition 1 (Contrastive Estimation)**.** Given a training set of positive samples $\mathcal{A}^+$ on which ϕ_{MLE} is defined and a negative distribution ν over $2^{\mathcal{Y}}$ (the power set of $\mathcal{Y}$), we call *Contrastive Estimation* the optimization problem

$$\text{Find } \boldsymbol{L} \in \operatorname*{argmax}_{\boldsymbol{L}\succeq 0} \phi_{\text{CE}}(\boldsymbol{L}) \tag{3}$$
$$\phi_{\text{CE}}(\boldsymbol{L}) \;\triangleq\; \phi_{\text{MLE}}(\boldsymbol{L}) - \mathbb{E}_{A\sim\nu}[\log \mathcal{P}_{\boldsymbol{L}}(A)],$$

where we write $\mathcal{P}_{\boldsymbol{L}}(A) \equiv \det(\boldsymbol{L}_A)/\det(\boldsymbol{L}+\boldsymbol{I})$.

The expectation can be approximated by drawing a set of samples $\mathcal{A}^-$ from ν, in which case ϕ_{CE} becomes[1]

$$\phi_{\text{CE}}(\boldsymbol{L}) = \tfrac{1}{|\mathcal{A}^+|}\sum_{A\in\mathcal{A}^+} \log\mathcal{P}_{\boldsymbol{L}}(A) - \tfrac{1}{|\mathcal{A}^-|}\sum_{A\in\mathcal{A}^-} \log\mathcal{P}_{\boldsymbol{L}}(A) \tag{4}$$

If $|\mathcal{A}^-| = 0$, the CE objective (3) reduces to ϕ_{MLE}.

Conversely, the basic objective function ϕ_{MLE} can be viewed as a sample-based approximation of the value $\mathbb{E}_{A\sim\mu}[\log \boldsymbol{P}_{\boldsymbol{L}}(A)]$, where μ is the true distribution generating the samples in $\mathcal{A}^+$.

Interestingly, another reformulation of (3) suggests an even broader class of DPP kernel learning: indeed, let y_A be $\frac{1}{|\mathcal{A}^+|}$ (resp. $-\frac{1}{|\mathcal{A}^-|}$) for $A \in \mathcal{A}^+$ (resp. $\mathcal{A}^-$), and define

$$\mathcal{A} = \{(y_A, A) : A\in\mathcal{A}^+\} \cup \{(y_A, A) : A \in \mathcal{A}^-\},$$

where the y_A should be viewed as belonging in $\{-1, 1\}$ with an additional normalization coefficient. Then, we can rewrite equation (4) in the following form

$$\phi_{\text{CE}}(\boldsymbol{L}) = \sum_{(y_A,A)\in\mathcal{A}} y_A \Big[\log\det \boldsymbol{L}_A - \det(\boldsymbol{L}+\boldsymbol{I})\Big]. \tag{5}$$

Formulation (5) suggests the use of a broader scope of continuous labels y_A; we do not cover this variation in the present work, but note that (5) permits the use of *weighted* samples in the learning process.

Remark 1. Compared to the traditional Noise Contrastive Estimation (NCE) approach, which requires full knowledge of the negative distribution, CE does not suffer any such limitation: we only require an estimate of $\mathbb{E}_\nu[\log \mathcal{P}_{\boldsymbol{L}}(A)]$.

[1] With a slight abuse of notation, we continue writing ϕ_{CE} despite the sample based approximation to $\mathbb{E}_{A\sim\nu}[\cdot]$.

Remark 2. Eq. (3) can be made to go to $+\infty$ with pathological negative samples (i.e. $\mathcal{P}_{\boldsymbol{L}}(A^-) = 0$); hence, choosing the negative distribution is a crucial concern for CE.

Indeed, to fully specify the CE problem one must first choose the negative distribution ν, or equivalently, choose a procedure to generate negative samples to obtain (4). We consider below two classes of distributions ν with considerably different ramifications: dynamic negatives and static negatives; their analysis is the focus of the next two sections.

3.2 Dynamic negatives

In most applications leveraging negative information (e.g., negative mining, GANs), negative samples evolve over time based on the state of the learned model. We call any ν that depends on the state of the model a *dynamic negative distribution*: at iteration k of the learning procedure with kernel estimate $\boldsymbol{L}_k$, we use a ν parametrized by $\boldsymbol{L}_k$.

More specifically, we focus on the setting where negative samples themselves are generated by the current DPP, with the goal of reducing overfitting. Given a positive sample A^+, we generate a negative A^- by replacing $i \in A^+$ with j that yields a high probability $\mathcal{P}_{\boldsymbol{L}_k}(A^+ \backslash \{i\} \cup \{j\})$ (Alg. 1). We generate the samples probabilistically rather than via mode maximization so that a sample A^+ can lead to different A^- negatives when we generate more negatives than positives.

Algorithm 1 Generate dynamic negative

Input: Positive sample A^+, current kernel $\boldsymbol{L}_k$
Sample $i \in A^+$ prop. to its empirical probability in $\mathcal{A}^+$
$A^- := A^+ \backslash \{i\}$
Sample j w.p. proportional to $\mathcal{P}_{\boldsymbol{L}_k}(A^- \cup \{j\})$
$A^- \leftarrow A^- \cup \{j\}$
return A^-

As ν evolves along with the kernel estimate $\boldsymbol{L}_k$, the second term of the objective function ϕ_{CE} acts as a moving target that must be continuously estimated during the learning procedure. For this reason, we choose to optimize ϕ_{CE} by a two-step procedure described in Alg. 2, similarly to an alternating maximization approach such as EM.

Algorithm 2 Optimizing dynamic CE

Input: Positive samples $\mathcal{A}^+$, initial kernel $\boldsymbol{L}_0$, maxIter.
$k \leftarrow 1$
while $k <$maxIter **and** not converged **do**
  $\mathcal{A}^- \leftarrow \textsc{GenerateDynamicNegatives}(\boldsymbol{L}_k, \mathcal{A}^+)$
  $\boldsymbol{L}_{k+1} \leftarrow \textsc{OptimizeCE}(\boldsymbol{L}_k, \mathcal{A}^+, \mathcal{A}^-)$
  $k \leftarrow k + 1$
end while
return $\boldsymbol{L}_k$

Note that this approach bears strong similarities with GANs, in which both the generator and discriminator evolve during training. Interestingly, the notion of dynamic negatives also appears in a discussion by Goodfellow [22], where they are used as a theoretical tool to analyze the difference between NCE and GANs.

Once the generated negative A^- has been used in an iteration of the optimization of ϕ_{CE}, it is less likely to be sampled again. Crucially, such dynamic negatives also avoid the problem alluded to in Remark 2, since by construction they have a non-zero probability under $\mathcal{P}_{\boldsymbol{L}_k}$ at iteration k.

3.3 Static negatives

Conversely, we can simplify the optimization problem by considering a *static* negative distribution: ν does not depend on the current kernel estimate.

A considerable theoretical advantage of static negatives over dynamic negatives lies in their simpler optimization problem: given a static ν, the optimization objective ϕ_{CE} does not evolve during training, and is amenable to a simple invocation of stochastic gradient descent [8].

Note, however, that such distributions may suffer from the fundamental theoretical issue in Rem. 2, and hence careful attention must be paid to ensure that the learning algorithm does not converge to a spurious optimum that assigns a probability $\boldsymbol{P_L}(A) = 0$ to $A \in \mathcal{A}^-$. In practice, we observed that the local nature of stochastic gradient ascent iterations was sufficient to avoid such behavior.

Let us now discuss two classical choices for fixed ν.

Product negatives. A common choice of negative distribution in other machine learning areas is the *product distribution*, which is the standard "noise" distribution used in NCE. It is defined by

$$\nu(A) = \prod\nolimits_{i \in A} \hat{p}(i) \prod\nolimits_{i \notin A} (1 - \hat{p}(i)) \tag{6}$$

where $\hat{p}(i)$ is the empirical probability of $\{i\}$ in $\mathcal{A}^+$. Although Mikolov et al. [35] report better results by raising the $\hat{p}$ to the power $\frac{3}{4}$, we did not observe any significant improvements when using exponentiated power distributions; for this reason, by *product negatives*, we always indicate the baseline distribution described by (6).

The product distribution is somewhat of mismatch for the DPP setting, as it lacks the crucial negative association property of DPPs, which is what enables them to model the repulsive interactions between similar items[2].

Explicit negatives. Alternatively, we may have explicit knowledge of a class of subsets that our model should *not* generate. For example, we might know that items i and j are very negatively correlated and hence unlikely to co-occur. Additionally, we may know via user feedback that some subsets generated by our model are inaccurate. We refer to negatives obtained using such outside information as *explicit negatives*.

A fundamental advantage of explicit negatives is that they allow us to incorporate prior knowledge and user feedback as part of the learning algorithm. The ability to incorporate such information, to our knowledge, is in itself a novel contribution to DPP learning.

Although such knowledge may be costly and/or only available at rare intervals, note that a form of continuous learning that would regularly update the state of our prior knowledge (and hence ν) would bring the explicit negative distribution into the realm of dynamic distributions, as described by Alg. 2.

4 Efficient learning and prediction

We now describe how the Contrastive Estimation problem for DPPs can be optimized efficiently.

In order to efficiently generate dynamic negatives, which rely on DPP conditioning, we additionally generalize the dual transformation leveraged in [37] to speed up basket-completion tasks with DPPs. This speed-up impacts the broader use of DPPs, outside of CE learning.

4.1 Optimizing ϕ_{CE}

We propose to optimize the CE problem by exploiting a low-rank factorization of the kernel, writing $\boldsymbol{L} = \boldsymbol{V}\boldsymbol{V}^\top$, where $\boldsymbol{V} \in \mathbb{R}^{M \times K}$ and $K \leq M$ is the rank of the kernel, which is fixed *a priori*.

This factorization ensures that the estimated kernel remains positive semi-definite, and furthermore enables us to leverage the low-rank computations derived in [18] and refined in [37]. Given the similar forms of the MLE and CE objectives, we leverage the traditional stochastic gradient ascent algorithm introduced by [18] to optimize (3). In the case of dynamic negatives, we re-generate the set $\mathcal{A}^-$ after each gradient step; note that less frequent updates are also possible if the negative generation algorithm is very costly.

[2]Specifically, DPPs belong to the family of *Strongly Rayleigh* measures, which have been shown to verify a broad range of negatively associated properties; we refer the interested reader to the fascinating body of work by [38, 5, 6, 2–4].

We furthermore augment ϕ_{CE} with a regularization term $R(\boldsymbol{V})$, defined as

$$R(\boldsymbol{V}) = \sum\nolimits_{i=1}^{M} \frac{1}{\mu_i} \|\boldsymbol{v}_i\|_2^2,$$

where μ_i is the number of occurrences of item i in the training set and $\boldsymbol{v}_i$ is the corresponding row vector of $\boldsymbol{V}$.

This regularization tempers the strength of $\|\boldsymbol{v}_i\|_2$, a term interpretable as to the popularity of item i [27, 19], based on its *empirical* popularity μ_i. Experimentally, we observe that adding $R(\boldsymbol{V})$ has a strong impact on the predictive quality of our model.

The reader may wonder if other approaches to DPP learning are also applicable to the CE problem.

Remark 3. Gradient ascent algorithms require that the estimate $\boldsymbol{L}$ be projected onto the space of positive semi-definite matrices; however, doing so can lead to almost-diagonal kernels [20] that cannot model negative interactions. Riemannian gradient ascent methods were considered, but deemed too computationally demanding by [32]. Furthermore, although it is tempting to generalize the fixed-point approach of Mariet & Sra [32] to ϕ_{CE}, the update rule that ensues does not admit a closed form solution, rendering it impractical (App. B).

The low-rank formulation allows us to apply CE (as well as NCE, as discussed in Section 5) to learn large datasets such as the Belgian retail supermarket dataset (described in Section 5) without prohibitive learning runtimes. We show below that by leveraging the idea described in [37], the low-rank formulation can also lead to additional speed ups during prediction.

4.2 Efficient conditioning for predictions

Dynamic negatives rely upon conditioning a DPP on a chosen sample A (see Alg. 1: $\mathcal{P}_{\boldsymbol{L}_k}(A^- \cup \{j\})$ can be efficiently computed for all j by a preprocessing step that conditions $\boldsymbol{L}_k$ on set A^-). For this reason, we now describe how low-rank DPP conditioning can be significantly sped up.

In [18], conditioning has a cost of $\mathcal{O}(K|\bar{A}|^2 + |A|^3)$, where $\bar{A} = \mathcal{Y} - A$. Since $|\mathcal{Y}| \gg |A|$ for many datasets, this represents a significant bottleneck for conditioning and computing next-item predictions for a set. We show here that through use of the dual transformation of the low-rank DPP kernel, we can reduce this complexity.

Proposition 1. *Given $A \subseteq \{1, \dots, M\}$ and a DPP of rank K parametrized by $\boldsymbol{V}$, where $\boldsymbol{L} = \boldsymbol{V}\boldsymbol{V}^\top$, we can derive the conditional marginal probabilities in the DPP parametrization $\boldsymbol{L}^A$ in $\mathcal{O}(K^3 + |A|^3 + K^2|A|^2 + |\bar{A}|K^2)$ time.*

Proof. As described in [19], we begin by computing the dual low-rank DPP kernel $\boldsymbol{C} = \boldsymbol{B}^\top \boldsymbol{B}$, where $\boldsymbol{B} = \boldsymbol{V}^\top$. We then compute

$$\boldsymbol{C}^A = \boldsymbol{B}^A (\boldsymbol{B}^A)^\top = \boldsymbol{Z}^A \boldsymbol{C} \boldsymbol{Z}^A,$$

with $\boldsymbol{Z}^A = \boldsymbol{I} - \boldsymbol{B}_A(\boldsymbol{B}_A^\top \boldsymbol{B}_A)^{-1} \boldsymbol{B}_A^\top$, and where $\boldsymbol{C}^A$ is the DPP kernel conditioned on the event that all of the elements in set A are observed, and $\boldsymbol{B}_A$ is the restriction of $\boldsymbol{B}$ to the rows and columns indexed by elements in A.

Computing $\boldsymbol{C}^A$ has a computational complexity of $\mathcal{O}(K^3 + |A|^3 + K^2|A|^2)$. Next, following [27], we eigendecompose $\boldsymbol{C}^A$ to compute the conditional (marginal) probability P_i of every possible item i in $\bar{A}$:

$$P_i = \sum_{n=1}^{K} \frac{\lambda_n}{\lambda_n + 1} \left(\frac{1}{\sqrt{\lambda_n}} \boldsymbol{b}_i^A \hat{\boldsymbol{v}}_n \right)^2$$

where $\boldsymbol{b}_i^A$ is the column vector from $\boldsymbol{B}^A$ for item i, λ_n is an eigenvalue of $\boldsymbol{C}^A$, and $\hat{\boldsymbol{v}}_n$ is the corresponding eigenvector.

The computational complexity for computing the eigendecomposition is $\mathcal{O}(K^3)$, and computing P_i for all items in $\bar{A}$ costs $\mathcal{O}(|\bar{A}|K^2)$. Therefore, we have an overall computational complexity of $\mathcal{O}(K^3 + |A|^3 + K^2|A|^2 + |\bar{A}|K^2)$ for computing next-item conditionals/predictions for the low-rank DPP using the dual kernel, which is significantly superior to the typical cost of $\mathcal{O}(K|\bar{A}|^2 + |A|^3)$. $\square$

Table 1: Mean MPR and precision values for LowRank, and baseline improvement on LowRank for other methods. Positive values indicate the relevant metric performs better than LowRank, and bold values indicate improvement over LowRank that lies outside the standard deviation estimate. Experiments were run 5 times, with a negative to positive ratio of $\frac{1}{2}$ and α set to its optimal LowRank value for the Amazon registries and to 1 for the Belgian dataset. The MPR improvement on the Belgian dataset in particular is significant (see Section 5.2 for details).

(a) Amazon Registries

Metric	LowRank	Improvement over LowRank		
		Dyn	Exp	NCE
MPR	70.50	**0.92 ± 0.56**	**0.68 ± 0.62**	**0.86 ± 0.55**
p@1	9.96	0.67 ± 0.75	0.58 ± 0.76	0.20 ± 1.75
p@5	25.36	**1.04 ± 0.82**	**0.78 ± 0.67**	0.67 ± 1.09
p@10	36.50	**1.39 ± 0.85**	**1.13 ± 0.79**	0.97 ± 1.18
p@20	51.22	**1.38 ± 0.97**	**1.28 ± 1.11**	**1.35 ± 1.20**

(b) Belgian dataset

LowRank	Improvement over LowRank	
	Exp	Dyn
79.62	**9.40 ± 0.28**	**9.38 ± 0.30**
13.23	-0.25 ± 0.39	-0.05 ± 0.25
21.94	-0.04 ± 0.27	0.05 ± 0.55
23.85	-0.57 ± 0.22	-0.61 ± 0.44
25.64	-0.22 ± 0.34	-0.31 ± 0.47

As in most cases $K \ll |\bar{A}|$, this represents a substantial improvement in the efficiency of DPP conditioning, and allows us to perform conditioning with complexity that is essentially linear in the size of the item catalog.

5 Experiments

We run experiments on two item recommendation datasets for DPP evaluation: the Amazon Baby Registry dataset [20], which has become a standard dataset for DPP modeling [32, 18] (described further in App. A), and the Belgian Retail Supermarket dataset[3], which contains 88,163 subsets, of a total of 16,470 unique items, collected as described in [10, 9]. We conduct an extensive experimental analysis on the smaller Amazon Registry dataset, and then use findings obtained from this analysis regarding appropriate hyperparameter settings to conduct experiments on the larger Belgian Retail dataset.

We compare the following Contrastive Estimation approaches:

- Exp: explicit negatives learned with CE. As to our knowledge there are no datasets with available explicit negative information, we generate approximations of explicit negatives by removing one item from a positive sample and replacing it with the least likely item (Additional details are provided in Appendix C).
- Dyn: dynamic negatives learned with CE.
- Prod: product negatives learned with CE.

As our work revolves around improving DPP performance, we focus on the two following baselines:

- NCE: Noise Contrastive Estimation with product negatives.
- LowRank: the standard low-rank DPP stochastic gradient ascent algorithm from [18].

NCE learns a model by contrasting $\mathcal{A}^+$ with negative samples drawn from a different but related "noisy" distribution p_n. Rather than explicitly minimizing the likelihood assigned to $\mathcal{A}^-$, NCE trains the model to distinguish between samples drawn from μ and samples drawn from p_n by maximizing the following conditional log-likelihood:

$$\phi_{\text{NCE}}(\boldsymbol{L}) = \sum_{A \in \mathcal{A}^+} \log P(A \in \mathcal{A}^+ \mid A) + \sum_{A \in \mathcal{A}^-} \log P(A \in \mathcal{A}^- \mid A). \tag{7}$$

NCE has gained popularity due to its ability to model distributions μ for which the normalization coefficient is unknown, and has been shown to be a powerful technique to improve submodular models for recommendation [43]. We learn the NCE objective with stochastic gradient ascent for our

[3] http://fimi.ua.ac.be/data/retail.pdf

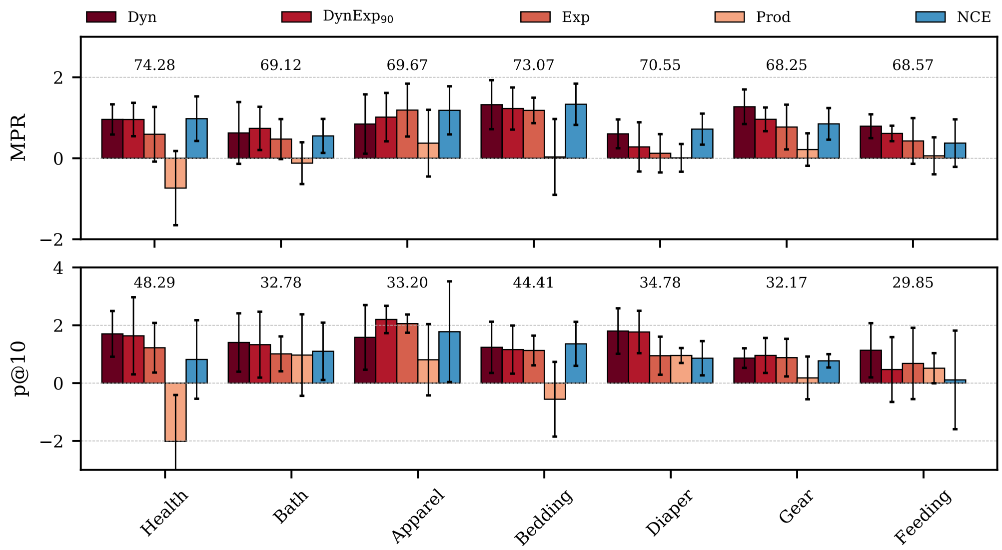

Figure 1: Absolute improvement of the MPR and precision@10 metrics for all methods on test data, compared to LowRank; error bars show standard deviation estimates. The numerical results indicate the baseline LowRank values for the metrics on each dataset. We set the negative to positive ratio to $\frac{1}{2}$ and set α to its optimal LowRank value; we use a low-rank matrix with $K = 30$ and minibatches of 200 positive samples. CE-based learning with dynamic or explicit negatives consistently improve the quality of the Dpp model, whereas NCE provides less significant improvements.

low-rank model, since $\nabla \log \Pr(A \in \mathcal{A}^* | A, \boldsymbol{V}\boldsymbol{V}^\top)$ is given by

$$\left(\epsilon^* - \frac{1}{1 + \frac{|\mathcal{A}^-|}{|\mathcal{A}^+|} p_n(A)/\mathcal{P}_{\boldsymbol{V}\boldsymbol{V}^\top}(A)}\right) \nabla_{\boldsymbol{V}} \log \mathcal{P}_{\boldsymbol{V}\boldsymbol{V}^\top}(A). \tag{8}$$

where $\epsilon^* = 1$ if $\mathcal{A}^* = \mathcal{A}^+$ and 0 otherwise.

The performance of all methods are compared using standard recommender system metrics: Mean Percentile Rank (MPR), and precision at k.

MPR is a recall-based metric which evaluates the model's predictive power by measuring how well it predicts the next item in a basket, and is a standard choice for recommender systems [24, 29]. Specifically, given a set A, let $p_{i,A} = \Pr(A \cup \{i\} \mid A)$. The percentile rank of an item i given a set A is defined as

$$\mathrm{PR}_{j,A} = \frac{\sum_{i' \notin A} \mathbb{1}(p_{i,A} \geq p_{i',A})}{|\mathcal{Y} \backslash A|} \times 100\%,$$

where $\mathcal{Y} \backslash A$ indicates those elements in the ground set $\mathcal{Y}$ that are not found in A, and we evaluate the MPR as

$$\frac{1}{|\mathcal{T}|} \sum_{A \in \mathcal{T}} \frac{1}{|A|} \sum_{i \in A} \mathrm{PR}_{i, A\backslash\{i\}}.$$

where $\mathcal{T}$ is the set of test instances. A MPR of 50 is equivalent to random selection; a MPR of 100 indicates that the model perfectly predicts the held out item. Precision at k measures how likely the model is to include the held-out item in its choice of top k completions; we evaluate it as

$$p@k = \frac{1}{|\mathcal{T}|} \sum_{A \in \mathcal{T}} \frac{1}{|A|} \sum_{i \in A} \mathbb{1}[\mathrm{rank}(i \mid A\backslash\{i\}) \leq k].$$

5.1 Amazon Baby Registries

For each category in the Amazon registries, all algorithms were tested on the 7 largest product categories.[4] 80% of subsets are used for training, and the remaining 20% served as test. All

[4]The smaller datasets, which comprise 7000 or less training samples (the "toys" dataset has exactly 7051), led to inconsistent results due to their size.

Table 2: Runtime to convergence, in seconds. Experiments were run on the feeding Amazon registry, with $\alpha = 1$, a negative to positive ratio of $\frac{1}{2}$, and $K = 30$.

METHOD	LOWRANK	EXP	DYN	NCE
RUNTIME	0.83 ± 0.54	2.69 ± 0.02	7.13 ± 0.28	27.59 ± 2.20

experimental results are averaged over 5 learning trials using different initial matrices, of rank $K = 30$.

In Table 1(a) and Figure 1, we compare the performance of the various learning algorithms. The regularization strength α is set to its optimal value for the LOWRANK algorithm, and we set $|\mathcal{A}^-|/|\mathcal{A}^+|= 1/2$. This allows us to compare the typical performance of the low-rank DPP algorithm to its "augmented" negative versions without hyper-parameter tuning. As PROD performs significantly worse than LOWRANK, we do not include it in further experiments.

Compared to traditional stochastic gradient ascent methods, algorithms that use inferred negatives perform (aside from PROD) better across all metrics on all datasets (additional results with other standard recommendation metrics are available in App. D.2). DYN and EXP provide consistent improvements compared to the other methods, whereas NCE shows a higher variance and slightly worse performance.

Note that improvements observed using DYN and EXP are larger than the loss in performance due to going from full-rank to low-rank kernels reported in [18].

We also investigated the performance of CE when mixing the source of the negatives: given the strong performance of DYN and EXP, we evaluated various ratios of dynamic to explicit negatives; as the performance of these mixed methods scales as expected between the two extremes, we only include the results for DYNEXP$_{90}$, which corresponds to 90% dynamic negatives and 10% explicit negatives. We see from Fig. 2 that the combination of dynamic and explicit negatives generally has a positive influence, as it performs better than the worst of either DYN or EXP. As such, it provides us with a more robust alternative.

Finally, we also compared all methods when tuning both the regularization α and the negative to positive ratio $\frac{|\mathcal{A}^-|}{|\mathcal{A}^+|}$, but did not see any significant improvements. As this suggests there is no need to do additional hyper-parameter tuning when using CE, we fix $\frac{|\mathcal{A}^-|}{|\mathcal{A}^+|} = \frac{1}{2}$ for all experiments.

Table 2 reports the average time to convergence for each method. As generating the dynamic negatives has a high complexity due to the cost of conditioning a DPP on different subsets, we see that DYN is 2.7x slower than EXP. LOWRANK is the fastest method, as it does not need to generate or process any negative samples. Finally, NCE is significantly more time-consuming than all the other methods, due to the additional complexity of computing the gradient.

We see from the results in Fig. 1 that DYN provides better performance than EXP on the MPR metric for several registry categories. On average, learning the DPP kernel via CE with dynamic or explicit negatives stand out as the best methods to obtain high predictive quality. Improvements in predictive quality provided by DYN come at the price of a marginally slower algorithm; this trade-off can be improved by adding a fraction of explicit negatives to the dynamic ones, as shown in Fig. 1.

5.2 Belgian Retail Dataset

As with the Amazon registries, we sample 80% of the Belgian Retail dataset for training + validation, and use the remaining 20% for testing. Following [17], we set K = 76 trait dimensions for the low-rank DPP, as the largest subset in this dataset is of size 76.

Once again, each experiment is run 5 times with different initializations. Following the results on the smaller Amazon registries dataset, we fix the ratio of negatives to positives to $\frac{|\mathcal{A}^-|}{|\mathcal{A}^+|} = 0.5$ and set $\alpha = 1$. Furthermore, given the significantly longer runtime of NCE, which is not compensated by consistent improvement in predictive quality, we do not evaluate it on the Belgian dataset.

Numerical results are provided in Table 1(b), and show that the negative methods give a significant improvement in performance over LOWRANK on MPR, with both DYN and EXP performing almost 10 points higher on the MPR metric. This is a striking improvement, compounded by small standard deviations confirming that these results are robust to different matrix initializations. Although we see no improvement on the precision@k metrics, MPR and precision@k are not always correlated: it is possible to improve the mean rank of all predictions without significantly affecting the top-k predictions (particularly for smaller values of k), since MPR scales with the size of the item catalog, while precision@k for a fixed k does not.

The gap between DYN and EXP observed on the Amazon dataset vanishes, suggesting that with the availability of more training data, EXP is preferable due to its more favorable runtime.

6 Conclusion and future work

We introduce the Contrastive Estimation (CE) optimization problem, which optimizes the difference of the traditional DPP log-likelihood and the expectation of the DPP model's log-likelihood under a *negative* distribution ν. This increases the DPP's fit to the data while simultaneously incorporating inferred or explicit domain knowledge into the learning procedure.

We show that CE lends itself to intuitively similar but theoretically different variants, depending on the choice of ν: a static negative distribution leads to significantly faster learning but allows spurious optima; conversely, allowing ν to evolve along with model parameters limits overfitting at the cost of a more complex optimization problem. Note that the question of optimizing dynamic CE is in of itself a theoretical problem worthy of independent study.

Additionally, we show that low-rank DPP conditioning complexity can be improved by a factor of M by leveraging the dual representation of the low-rank kernel. This not only improves prediction speed on a trained model, but allows for more efficient dynamic negative generation.

Experimentally, we show that CE with dynamic and explicit negatives provide comparable and significant improvements in the predictive performance of DPPs.

Our analysis also raises both theoretical and practical questions: in particular, a key component of future work lies in better understanding how explicit domain knowledge can be combined with dynamic and static negatives. Furthermore, the CE formulation in Eq. (5) suggests the possibility of using continuous labels to reflect the more general concept of weighted samples within Contrastive Estimation.

References

[1] Affandi, R., Fox, E., Adams, R., and Taskar, B. Learning the parameters of Determinantal Point Process kernels. In *ICML*, 2014.

[2] Borcea, Julius and Brändén, Petter. The lee-yang and pólya-schur programs. i. linear operators preserving stability. *Inventiones mathematicae*, 177(3):541–569, 2009.

[3] Borcea, Julius and Brändén, Petter. The lee-yang and pólya-schur programs. ii. theory of stable polynomials and applications. *Communications on Pure and Applied Mathematics*, 62(12): 1595–1631, 2009.

[4] Borcea, Julius and Brändén, Petter. Multivariate pólya–schur classification problems in the weyl algebra. *Proceedings of the London Mathematical Society*, 101(1):73–104, 2010.

[5] Borcea, Julius, Brändén, Petter, and Liggett, Thomas. Negative dependence and the geometry of polynomials. *Journal of the American Mathematical Society*, 22(2):521–567, 2009.

[6] Borcea, Julius, Brändén, Petter, and Shapiro, Boris. Classification of hyperbolicity and stability preservers: the multivariate weyl algebra case. *arXiv preprint math.CA/0606360*, 2009.

[7] Borodin, Alexei. Determinantal point processes. *arXiv:0911.1153*, 2009.

[8] Bottou, Léon. On-line learning in neural networks. chapter On-line Learning and Stochastic Approximations, pp. 9–42. Cambridge University Press, 1998.

[9] Brijs, Tom. Retail market basket data set. In *Workshop on Frequent Itemset Mining Implementations (FIMI'03)*, 2003.

[10] Brijs, Tom, Swinnen, Gilbert, Vanhoof, Koen, and Wets, Geert. Using association rules for product assortment decisions: A case study. In *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 254–260. ACM, 1999.

[11] Canévet, Olivier and Fleuret, Francois. Efficient Sample Mining for Object Detection. In *Proceedings of the 6th Asian Conference on Machine Learning (ACML)*, JMLR: Workshop and Conference Proceedings, 2014.

[12] Chao, Wei-Lun, Gong, Boqing, Grauman, Kristen, and Sha, Fei. Large-margin determinantal point processes. In *Uncertainty in Artificial Intelligence (UAI)*, 2015.

[13] Chen, Long, Yuan, Fajie, Jose, Joemon M., and Zhang, Weinan. Improving negative sampling for word representation using self-embedded features. *CoRR*, abs/1710.09805, 2017.

[14] Decreusefond, Laurent, Flint, Ian, Privault, Nicolas, and Torrisi, Giovanni Luca. Determinantal point processes, 2015.

[15] Djolonga, Josip, Tschiatschek, Sebastian, and Krause, Andreas. Variational inference in mixed probabilistic submodular models. In Lee, D. D., Sugiyama, M., Luxburg, U. V., Guyon, I., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 29*, pp. 1759–1767. Curran Associates, Inc., 2016.

[16] Dupuy, Christophe and Bach, Francis. Learning determinantal point processes in sublinear time, 2016.

[17] Gartrell, Mike, Paquet, Ulrich, and Koenigstein, Noam. Bayesian low-rank determinantal point processes. In Sen, Shilad, Geyer, Werner, Freyne, Jill, and Castells, Pablo (eds.), *Proceedings of the 10th ACM Conference on Recommender Systems*, pp. 349–356. ACM, 2016.

[18] Gartrell, Mike, Paquet, Ulrich, and Koenigstein, Noam. Low-rank factorization of determinantal point processes. In *AAAI*, 2017.

[19] Gillenwater, J. *Approximate Inference for Determinantal Point Processes*. PhD thesis, University of Pennsylvania, 2014.

[20] Gillenwater, J., Kulesza, A., Fox, E., and Taskar, B. Expectation-Maximization for learning Determinantal Point Processes. In *NIPS*, 2014.

[21] Goodfellow, Ian, Pouget-Abadie, Jean, Mirza, Mehdi, Xu, Bing, Warde-Farley, David, Ozair, Sherjil, Courville, Aaron, and Bengio, Yoshua. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pp. 2672–2680. Curran Associates, Inc., 2014.

[22] Goodfellow, Ian J. On distinguishability criteria for estimating generative models, 2014.

[23] Gutmann, Michael U. and Hyvärinen, Aapo. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *J. Mach. Learn. Res.*, 13: 307–361, February 2012. ISSN 1532-4435.

[24] Hu, Yifan, Koren, Yehuda, and Volinsky, Chris. Collaborative filtering for implicit feedback datasets. In *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, 2008.

[25] Krause, Andreas, Singh, Ajit, and Guestrin, Carlos. Near-optimal sensor placements in Gaussian processes: theory, efficient algorithms and empirical studies. *JMLR*, 9:235–284, 2008.

[26] Kulesza, A. *Learning with Determinantal Point Processes*. PhD thesis, University of Pennsylvania, 2013.

[27] Kulesza, A. and Taskar, B. *Determinantal Point Processes for machine learning*, volume 5. Foundations and Trends in Machine Learning, 2012.

[28] Lavancier, Frédéric, Møller, Jesper, and Rubak, Ege. Determinantal point process models and statistical inference. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(4):853–877, 2015.

[29] Li, Yanen, Hu, Jia, Zhai, ChengXiang, and Chen, Ye. Improving one-class collaborative filtering by incorporating rich user information. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, 2010.

[30] Lin, H. and Bilmes, J. Learning mixtures of submodular shells with application to document summarization. In *Uncertainty in Artificial Intelligence (UAI)*, 2012.

[31] Macchi, O. The coincidence approach to stochastic point processes. *Adv. Appl. Prob.*, 7(1), 1975.

[32] Mariet, Zelda and Sra, Suvrit. Fixed-point algorithms for learning determinantal point processes. In *ICML*, 2015.

[33] Mariet, Zelda and Sra, Suvrit. Diversity networks. *Int. Conf. on Learning Representations (ICLR)*, 2016.

[34] Mariet, Zelda and Sra, Suvrit. Kronecker Determinantal Point Processes. In *NIPS*, 2016.

[35] Mikolov, Tomas, Sutskever, Ilya, Chen, Kai, Corrado, Greg S, and Dean, Jeff. Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. Curran Associates, Inc., 2013.

[36] Mnih, Andriy and Teh, Yee Whye. A fast and simple algorithm for training neural probabilistic language models. In *In Proceedings of the International Conference on Machine Learning*, 2012.

[37] Osogami, Takayuki, Raymond, Rudy, Goel, Akshay, Shirai, Tomoyuki, and Maehara, Takanori. Dynamic determinantal point processes. In *AAAI*, 2018.

[38] Pemantle, Robin. Towards a theory of negative dependence. *Journal of Mathematical Physics*, 41(3):1371–1390, 2000.

[39] Shrivastava, Abhinav, Gupta, Abhinav, and Girshick, Ross. Training region-based object detectors with online hard example mining. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[40] Smith, Noah A. and Eisner, Jason. Guiding unsupervised grammar induction using contrastive estimation. In *In Proc. of IJCAI Workshop on Grammatical Inference Applications*, pp. 73–82, 2005.

[41] Smith, Noah A. and Eisner, Jason. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pp. 354–362. Association for Computational Linguistics, 2005.

[42] Sung, Kah Kay. *Learning and Example Selection for Object and Pattern Detection*. PhD thesis, Massachusetts Institute of Technology, 1996.

[43] Tschiatschek, Sebastian, Djolonga, Josip, and Krause, Andreas. Learning probabilistic submodular diversity models via noise contrastive estimation. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, pp. 770–779, 2016.

[44] Urschel, John, Brunel, Victor-Emmanuel, Moitra, Ankur, and Rigollet, Philippe. Learning determinantal point processes with moments and cycles. In *Proceedings of the 34th International Conference on Machine Learning, ICML*, pp. 3511–3520, 2017.

[45] Zhang, Cheng, Kjellström, Hedvig, and Mandt, Stephan. Stochastic learning on imbalanced data: Determinantal point processes for mini-batch diversification. *CoRR*, abs/1705.00607, 2017.

A Amazon Baby Registries description

Table 3: Numerical details for the Amazon Baby registries dataset.

REGISTRY	M	TRAIN SIZE	TEST SIZE
HEALTH	62	5278	1320
BATH	100	5510	1377
APPAREL	100	6482	1620
BEDDING	100	7119	1780
DIAPER	100	8403	2101
GEAR	100	7089	1772
FEEDING	100	10,090	2522

B Contrastive Estimation with the Picard iteration

Letting $\beta = |\mathcal{A}^+| - |\mathcal{A}^-| \geq 0$ and writing $\boldsymbol{U}_A$ as the $M \times |A|$ indicator matrix such that $\boldsymbol{L}_A = \boldsymbol{U}_A^\top \boldsymbol{L} \boldsymbol{U}_A$, we have

$$\phi(\boldsymbol{L}) \propto \underbrace{-\beta \log\det(\boldsymbol{I}+\boldsymbol{X}) + \sum_{A\in\mathcal{A}^+} \log\det(\boldsymbol{U}_A^\top \boldsymbol{X}^{-1}\boldsymbol{U}_A)}_{f \text{ convex}}$$
$$\underbrace{+\beta \log\det(\boldsymbol{X}) - \sum_{A\in\mathcal{A}^-} \log\det(\boldsymbol{U}_A^\top \boldsymbol{X}^{-1}\boldsymbol{U}_A)}_{g \text{ concave}}$$

where the convexity/concavity results follow immediately from [32, Lemma 2.3]. Then, the update rule $\nabla f(\boldsymbol{L}_{k+1}) = -\nabla g(\boldsymbol{L}_k)$ requires

$$\beta \boldsymbol{L}_{k+1} + \sum_{A\,\in\mathcal{A}^-} \boldsymbol{L}_{k+1}\boldsymbol{U}_A(\boldsymbol{U}_A^\top \boldsymbol{L}_{k+1}\boldsymbol{U}_A)^{-1}\boldsymbol{U}_A^\top \boldsymbol{L}_{k+1}$$
$$\leftarrow \beta(\boldsymbol{I}+\boldsymbol{L}_k^{-1})^{-1} + \sum_{A\in\mathcal{A}^+} \boldsymbol{L}_k\boldsymbol{U}_A(\boldsymbol{U}_A^\top \boldsymbol{L}_k\boldsymbol{U}_A)^{-1}\boldsymbol{U}_A^\top \boldsymbol{L}_k$$

which cannot be evaluated due to the $\sum_{A\in\mathcal{A}^-}$ term.

C Approximating explicit negatives

We use the empirical marginal distribution of items in the positive samples as a basis for approximating an explicit negative. For a given positive set A^+, we sample two items $i, j \in A^+$, and replace j with the item $k \notin A^+$ such that the empirical probability of observing items i and k in the same set is unlikely.

Algorithm 3 Approximate explicit negative

input: Positive sample A^+
Sample $i \neq j \in A^+$ w.p. $p_i \propto \widehat{P}(\{i\})$
Sample $k \notin A^+$ w.p. $p_k \propto 1 - \widehat{P}(\{i,k\})$.
return $(A^+ \backslash \{j\}) \cup \{k\}$

This allows us to approximate true explicit negatives, as we use the empirical data to derive "implausible" sets. Note, however, that when using such negatives, we have no guarantee that objective function will be well behaved, as opposed to the theoretically grounded use of dynamic negatives.

D Additional experimental results

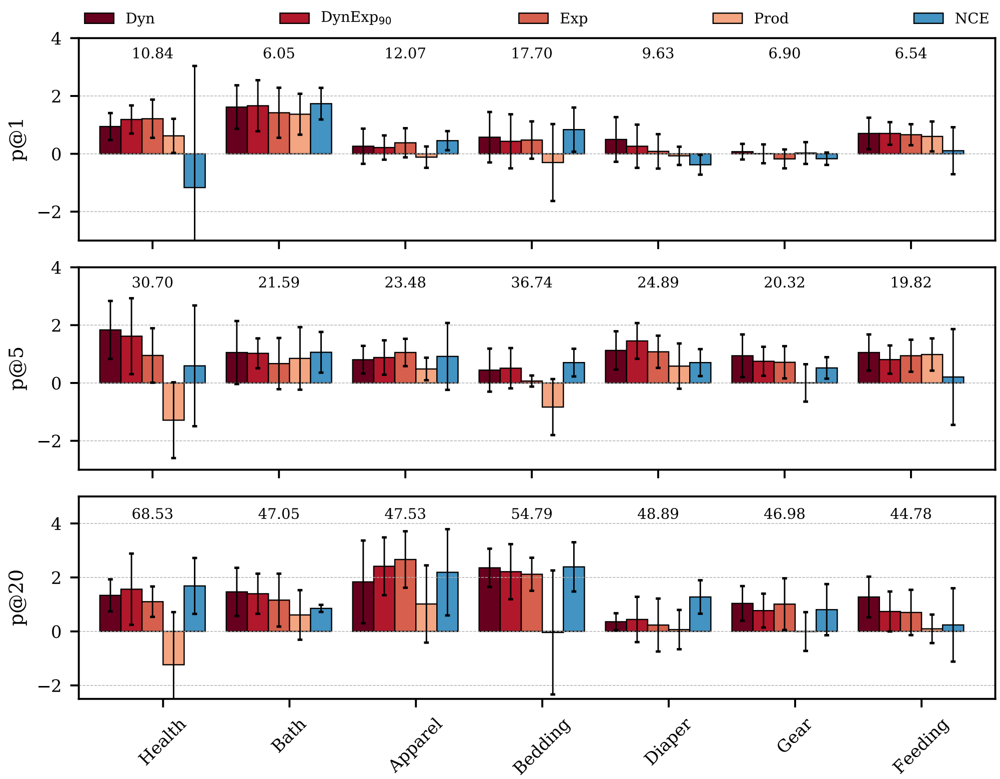

Figure 2: Test precisions@k for all methods using a negative to positive ratio of $1/2$ and the optimal α value for LOWRANK on the baby registries datasets.