Learning Determinantal Point Processes by Sampling Inferred Negatives

Zelda Mariet
Massachusetts Institute of Technology
zelda@csail.mit.edu

Mike Gartrell
Criteo AI Lab
m.gartrell@criteo.com

Suvrit Sra
Massachusetts Institute of Technology
suvrit@mit.edu

Abstract

Determinantal Point Processes (DPPs) have attracted significant interest in machine learning due to their ability to elegantly and tractably model the balance between quality and diversity of sets. We consider here the key task of learning DPPs from data; 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. Experimentally, we show that CE learning delivers a considerable improvement in DPP predictive performance.

1 Introduction and related work

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 $A \subseteq \mathcal{Y}$,

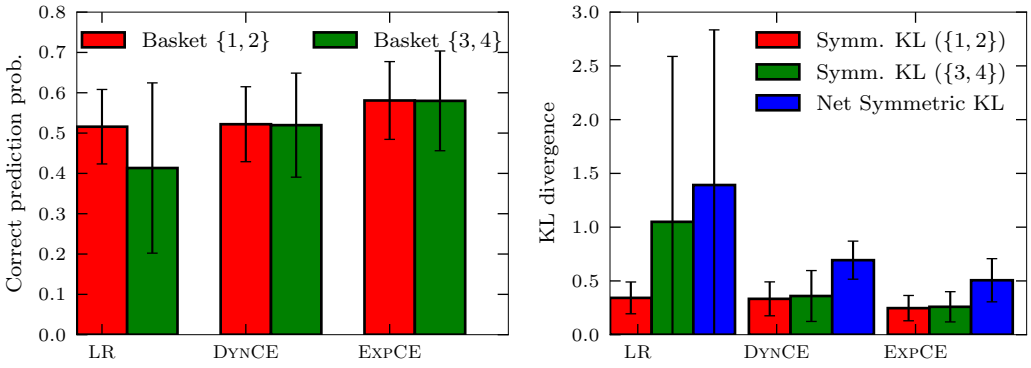
$$\Pr(A) \propto \det(\boldsymbol{L}_A), \tag{1}$$

where $\boldsymbol{L}_A = [\boldsymbol{L}_{ij}]_{i,j\in A}$ is the submatrix of $\boldsymbol{L}$ indexed by A. First introduced to model fermion behavior by Macchi [23], DPPs have gained popularity due to their elegant balancing of quality and subset diversity. DPPs are studied both for their theoretical properties [19, 2, 1, 18, 12, 8, 20] and their machine learning applications: object retrieval [1], summarization [22, 6], sensor placement [17], recommender systems [10], neural network compression [25], and minibatch selection [34].

The key object defining a DPP is its kernel matrix $\boldsymbol{L}$. This matrix may be fixed *a priori* using domain knowledge [2] or learned from observations using maximum likelihood estimation (MLE) [13, 24]. However, while fitting observed subsets well, MLE for DPPs may also assign high likelihoods to unobserved subsets far from the underlying generative distribution [6]. Such confusable modes reduce the quality of the learned model, hurting predictions (see Figure 1).

These observations motivate us to investigate DPP-generated samples with added perturbations as *negatives*, which we incorporate into the learning task to improve the modeling power of DPPs. As there is no closed form way to generate such idealized negatives, we approximate them via an external "negative distribution". Aside from [33, 9], little attention has been given to using negative

32nd Conference on Neural Information Processing Systems (NIPS 2018), Montréal, Canada.



(a) Probabilities of making the right prediction

(b) KL divergences

Figure 1: Results on a toy dataset generated by replicating the baskets {1, 2} and {3, 4} 1000 times each. We train each model to convergence, then compute the next-item probabilities for each pair, along with the symmetric KL divergence over areas of shared support between predictive and empirical next-item distributions. Experiments were run 10 times, with $|\mathcal{A}^+|/|\mathcal{A}^-|$ set to the optimal value for each model; α is set to its optimal LR value. LR (low-rank DPP) assigns high predictive probabilities to modes for incorrect predictions, resulting in higher symmetric KL divergence and high variances. The DYN and EXP methods we introduce reduce this issue, resulting in predictive distributions closer to the true distribution and smaller variances.

samples to learn richer subset-selection models. Nonetheless, leveraging negative information is widely used in other applications. In object detection, negative mining trains the model on its false positives [32, 5, 29]. In language modeling, Noise Contrastive Estimation (NCE) [15] was first applied in [27] and has been instrumental in Word2Vec [26]. An alternate approach within submodular language models was introduced as Contrastive Estimation in [30, 31].

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.

2 Learning DPPs with negative samples

Motivated by the similarities between DPP learning and crucial structured prediction problems in other ML 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 discussed in [30].

2.1 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.:

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

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 $2^{\mathcal{Y}}$, we call *Contrastive Estimation* the problem

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

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: ϕ_{CE} then becomes[1]

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

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

Remark 1. CE is a non-convex optimization problem, and thus admits the same guarantees as DPP MLE learning when learned using Stochastic Gradient Ascent with decreasing step sizes; however, the convergence rate will depend on the choice of ν.

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 and static negatives; their analysis is the focus of the next two sections.

2.2 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.

As ν evolves along with $\boldsymbol{L}_k$, the second term of 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.

Note that this approach bears strong similarities with GANs, in which both the generator and discriminator evolve during training (dynamic negatives also appear in a discussion by Goodfellow [14] as a theoretical tool to analyze the difference between NCE and GANs). Once a negative A^- has been used in an iteration of the optimization of ϕ_{CE}, it is less likely to be sampled again.[2]

2.3 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 lies in the simpler optimization problem: the optimization objective ϕ_{CE} does not evolve during training, and is amenable to a simple invocation of stochastic gradient descent [4].

Theorem 1. *Let ν be a static distribution over $2^{\mathcal{Y}}$ and let $k > 0$ be such that $k \geq \max\{|S|\colon S \in \mathcal{A}^+ \cup \text{supp}(\nu)\}$. Let K be a bounded subspace of all $|\mathcal{Y}|\times|\mathcal{Y}|$ positive semi-definite matrices of rank k. Projected stochastic gradient ascent applied to the CE objective with negative distribution ν and space K with step sizes η_i such that $\sum \eta_i = \infty$, $\sum \eta_i^2 < \infty$ will converge to a critical point.*

Unlike dynamic negatives, careful attention must be paid to ensure that the learning algorithm does not converge to a spurious optimum that assigns $\boldsymbol{P}_{\boldsymbol{L}}(A) = 0$ to $A \in \mathcal{A}^-$. In practice, we observed that the local nature of stochastic gradient ascent iterations was sufficient to avoid such behavior.

We may have prior knowledge of a class of subsets that our model should *not* generate. For example, we might know that items i and j are negatively correlated and hence unlikely to co-occur. We may also learn via user feedback that some generated subsets 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 during learning.

2.4 Efficient learning and prediction

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 $\boldsymbol{V} \in \mathbb{R}^{|\mathcal{Y}|\times K}$ and $K \leq |\mathcal{Y}|$ is the rank of the kernel, which is fixed *a priori*. This factorization ensures that the estimated kernel remains positive semi-definite, and enables us to leverage the low-rank computations derived in [11] and refined in [28]. Given the similar forms of the MLE and CE objectives, we use the traditional stochastic gradient ascent algorithm introduced by [11] to optimize (3). In the case of dynamic negatives, we re-generate $\mathcal{A}^-$ after each gradient step; less frequent updates are also possible if the negative generation algorithm is very costly.

The low-rank formulation allows us to apply CE (and the NCE baseline) to large datasets without prohibitive runtimes. We show in App. C that this formulation can also lead to additional speed

[2] If A^- happens to be a false negative (i.e. appears in $\mathcal{A}^+$), A^- will be comparatively sampled more frequently as a positive, and so will contribute on average as a positive sample. Additional precautions such as the ones mentioned in [3] can also be leveraged if necessary.

(a) UK dataset

Metric	LR	Improvement over LR: EXP	Improvement over LR: DYN
MPR	80.07	**3.75 ± 0.16**	**3.74 ± 0.16**
AUC	0.57297	**0.41465 ± 0.01334**	**0.41467 ± 0.01339**

(b) Belgian dataset

LR	Improvement over LR: EXP	Improvement over LR: DYN
79.62	**9.40 ± 0.28**	**9.38 ± 0.30**
0.6159	**0.3705 ± 3.287e-4**	**0.3707 ± 3.140e-4**

Table 1: Results over the UK and Belgian datasets. Both explicit and dynamic CE obtain significant improvements in MPR and AUC metrics, confirming that CE learning enhances recommender value of the model and its ability to distinguish data drawn from the target distribution from fake samples.

ups during prediction. We augment ϕ_{CE} with a regularization term $R(\boldsymbol{V})$ as in [11], defined as $R(\boldsymbol{V}) = \alpha \sum_{i=1}^{|\mathcal{Y}|} \frac{1}{\mu_i} \|\boldsymbol{v}_i\|_2^2$, where μ_i counts the occurrences of item i in the training set, $\boldsymbol{v}_i$ is i-th row vector of $\boldsymbol{V}$, and $\alpha > 0$ is a tunable hyperparameter. This tempers the strength of $\|\boldsymbol{v}_i\|_2$, a term interpretable as to the popularity of item i [19, 12] based on its *empirical* popularity μ_i.

3 Experiments

We run next-item prediction and AUC-based classification experiments on two recommendation datasets of purchased shopping baskets: the UK retail dataset [7] and the large Belgian Retail Supermarket dataset.[3] We compare explicit (EXP) and dynamic CE (DYN) to standard low-rank (LR) learning [11]. We saw that one iteration of NCE [33] required nearly 11 hours on the Belgian dataset (compared to 5 minutes for one iteration of CE); see Appendix E for details. For this reason, we remove NCE as a baseline from all experiments, as it is not feasible in the general case. As to our knowledge there are no datasets with explicit negative information, we generate approximations of explicit negatives by replacing one item in each positive sample by the least likely item (Alg. 3).

DPPs are a commonly accepted modeling tool for recommender systems that require balancing diversity and quality. For this reason, we focus on evaluating the performance of all methods are compared using the standard MPR (Mean Percentile Rank) metric (see App. D) for recommender systems [16, 21]. An MPR of 50 is equivalent to random selection; a MPR of 100 indicates that the model perfectly predicts the held out item. We also evaluate the discriminative power of each model using the AUC metric. For this task, we generate a set of fake subsets uniformly at random, then compute the AUC for the model's ability to discriminate between true and fake subsets. Following [10], for both the UK and Belgian datasets, we set the rank K of the kernel to be the size of the largest subset in the dataset (K=100 for the UK dataset, K=76 for the Belgian dataset); this optimizes memory costs while still modeling all ground-truth subsets. Based on our results on the smaller Amazon dataset (App. F), we fix $|\mathcal{A}^-|/|\mathcal{A}^+| = 0.5$ and $\alpha = 1$.

Tables 1 (a) and (b) summarize our results; the DYN and EXP negative methods show a striking MPR and AUC improvement over LR. These results suggest that for larger datasets, CE is effective at improving predictive and discriminative power for DPPs.

4 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.

CE lends itself to intuitively similar but theoretically different variants, depending on the choice of ν: a static ν 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. Optimizing dynamic CE is in of itself a theoretical problem worthy of independent study.

Experimentally, we show that CE with dynamic and explicit negatives provide comparable, significant improvements in the predictive performance of DPPs, as well as on the learned DPP's ability to discriminate between real and randomly generated subsets.

Acknowledgements. This work was partially supported by a Criteo Faculty Research Award, and NSF-IIS-1409802.

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

References

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

[2] Alexei Borodin. Determinantal Point Processes. *arXiv:0911.1153*, 2009.

[3] Avishek Bose, Huan Ling, and Yanshuai Cao. Adversarial contrastive estimation. *arXiv preprint arXiv:1805.03642*, 2018.

[4] Léon Bottou. On-line learning in neural networks. Cambridge University Press, 1998.

[5] Olivier Canévet and Francois Fleuret. Efficient Sample Mining for Object Detection. In *ACML*, JMLR: Workshop and Conference Proceedings, 2014.

[6] Wei-Lun Chao, Boqing Gong, Kristen Grauman, and Fei Sha. Large-margin Determinantal Point Processes. In *Uncertainty in Artificial Intelligence (UAI)*, 2015.

[7] D Chen. Data mining for the online retail industry: A case study of rfm model-based customer segmentation using data mining. *Journal of Database Marketing and Customer Strategy Management*, 19(3), August 2012.

[8] Laurent Decreusefond, Ian Flint, Nicolas Privault, and Giovanni Luca Torrisi. Determinantal Point Processes, 2015.

[9] Josip Djolonga, Sebastian Tschiatschek, and Andreas Krause. Variational inference in mixed probabilistic submodular models. In *NIPS*. Curran Associates, Inc., 2016.

[10] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. Bayesian low-rank Determinantal Point Processes. In Shilad Sen, Werner Geyer, Jill Freyne, and Pablo Castells, editors, *RecSys*. ACM, 2016.

[11] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. Low-rank factorization of Determinantal Point Processes. In *AAAI*, 2017.

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

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

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

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

[16] Yifan Hu, Yehuda Koren, and Chris Volinsky. Collaborative filtering for implicit feedback datasets. In *ICDM*, 2008.

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

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

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

[20] Frédéric Lavancier, Jesper Møller, and Ege Rubak. Determinantal Point Process models and statistical inference. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 77(4), 2015.

[21] Yanen Li, Jia Hu, ChengXiang Zhai, and Ye Chen. Improving one-class collaborative filtering by incorporating rich user information. In *CIKM*, 2010.

[22] H. Lin and J. Bilmes. Learning mixtures of submodular shells with application to document summarization. In *UAI*, 2012.

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

[24] Zelda Mariet and Suvrit Sra. Fixed-point algorithms for learning Determinantal Point Processes. In *ICML*, 2015.

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

[26] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*. 2013.

[27] Andriy Mnih and Yee Whye Teh. A fast and simple algorithm for training neural probabilistic language models. In *ICML*, 2012.

[28] Takayuki Osogami, Rudy Raymond, Akshay Goel, Tomoyuki Shirai, and Takanori Maehara. Dynamic Determinantal Point Processes. In *AAAI*, 2018.

[29] Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. Training region-based object detectors with online hard example mining. In *CVPR*, 2016.

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

[31] Noah A. Smith and Jason Eisner. Contrastive estimation: Training log-linear models on unlabeled data. In *ACL*, ACL '05, 2005.

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

[33] Sebastian Tschiatschek, Josip Djolonga, and Andreas Krause. Learning probabilistic submodular diversity models via noise contrastive estimation. In *AISTATS*, 2016.

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

A 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 [24, 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.

B Algorithms

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^-

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**
  $\mathcal{A}^- \leftarrow$ GENERATEDYNAMICNEGATIVES($\boldsymbol{L}_k$, $\mathcal{A}^+$)
  $\boldsymbol{L}_{k+1} \leftarrow$ OPTIMIZECE($\boldsymbol{L}_k, \mathcal{A}^+, \mathcal{A}^-$)
end while
return $\boldsymbol{L}_k$

Algorithm 3 Approximate explicit negative generation

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 $(A^+ \backslash \{j\}) \cup \{k\}$

C 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 [11], 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 this complexity can be brought down significantly.

Proposition 1. *Given $A \subseteq \{1, \ldots, M\}$ 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 only $\mathcal{O}(K^3 + |A|^3+K^2|A|^2+|\bar{A}|K^2)$ time.*

As in most cases $K \ll |\bar{A}|$, this represents a substantial improvement, allowing us condition in time essentially linear in the size of the item catalog.

D Mean Percentile Rank (MPR)

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 [16, 21].

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

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

The MPR is then computed as

$$\text{MPR} = \frac{1}{|\mathcal{T}|} \sum_{A \in \mathcal{T}} \text{PR}_{i,A\backslash\{i\}}$$

where $\mathcal{T}$ is the set of test instances and i is a randomly selected element in each set A. An MPR of 50 is equivalent to random selection; a MPR of 100 indicates that the model perfectly predicts the held out item.

E Noise Contrastive Estimation (NCE)

NCE learns a model by contrasting $\mathcal{A}^+$ with negatives drawn from a "noisy" distribution p_n, training the model to distinguish between sets drawn from μ and sets drawn from p_n. NCE has gained popularity due to its ability to model distributions μ with untractable normalization coefficients, and has been shown to be a powerful technique to improve submodular recommendation models [33]. NCE learns by maximizing the following conditional log-likelihood:

$$\begin{aligned}\phi_{\text{NCE}}(\boldsymbol{L}) = &\sum_{A \in \mathcal{A}^+} \log P(A \in \mathcal{A}^+ \mid A) \\ &+ \sum_{A \in \mathcal{A}^-} \log P(A \in \mathcal{A}^- \mid A). \end{aligned} \tag{5}$$

In our experiments, we learn the NCE objective with stochastic gradient ascent for our low-rank model, since $\nabla \log \Pr(A \in \mathcal{A}^* | A, \boldsymbol{V}\boldsymbol{V}^\top)$ is given by

$$\Big(\epsilon^* - \Big(1 + \frac{|\mathcal{A}^-|}{|\mathcal{A}^+|} \frac{p_n(A)}{\mathcal{P}_{\boldsymbol{V}\boldsymbol{V}^\top}(A)}\Big)^{-1}\Big) \nabla_{\boldsymbol{V}} \log \mathcal{P}_{\boldsymbol{V}\boldsymbol{V}^\top}(A). \tag{6}$$

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

F Amazon Baby registries experiments

In Tab. 3(a), we compare the performance of the various algorithms with rank $K = 30$. The regularization strength α is set to its optimal value for the LR algorithm, and $|\mathcal{A}^-|/|\mathcal{A}^+| = 1/2$. This allows us to compare the LR algorithm to its "augmented" negative versions without hyper-parameter tuning. As PROD performs much worse than LR, it is not included in further experiments.

We evaluate the precision at k as

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

Table 2: Description of 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

Table 3: MPR, p@k, and AUC values for LR, and baseline improvement over LR for other methods. Positive values indicate the algorithm performs better than LR, and bold values indicate improvement over LR that lies outside the standard deviation. Experiments were run 5 times, with $|\mathcal{A}^+|/|\mathcal{A}^-| = \frac{1}{2}$; α is set to its optimal LR value.

		Improvement over LR		
Metric	LR	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**
AUC	0.630	**0.027 ± 0.017**	**0.026 ± 0.016**	0.009 ± 0.017

Compared to traditional SGA methods, algorithms that use inferred negatives perform (PROD excepted) better across all metrics and datasets. DYN and EXP provide consistent improvements compared to the other methods, whereas NCE shows a higher variance and slightly worse performance. 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 [11].

Finally, we also compared all methods when tuning both the regularization α and the negative to positive ratio $|\mathcal{A}^-|/|\mathcal{A}^+|$, 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.