k-DPPs: Fixed-Size Determinantal Point Processes

Alex Kulesza KULESZA@CIS.UPENN.EDU
Ben Taskar TASKAR@CIS.UPENN.EDU
University of Pennsylvania, Department of Computer and Information Science, Philadelphia, PA 19104 USA

Abstract

Determinantal point processes (DPPs) have recently been proposed as models for set selection problems where diversity is preferred. For example, they can be used to select diverse sets of sentences to form document summaries, or to find multiple non-overlapping human poses in an image. However, DPPs conflate the modeling of two distinct characteristics: the *size* of the set, and its *content*. For many realistic tasks, the size of the desired set is known up front; e.g., in search we may want to show the user exactly ten results. In these situations the effort spent by DPPs modeling set size is not only wasteful, but actually introduces unwanted bias into the modeling of content. Instead, we propose the k-DPP, a conditional DPP that models only sets of cardinality k. In exchange for this restriction, k-DPPs offer greater expressiveness and control over content, and simplified integration into applications like search. We derive algorithms for efficiently normalizing, sampling, and marginalizing k-DPPs, and propose an experts-style algorithm for learning combinations of k-DPPs. We demonstrate the usefulness of the model on an image search task, where k-DPPs significantly outperform MMR as judged by human annotators.

1. Introduction

Determinantal point processes (DPPs) have recently been proposed as models for set selection problems where diversity is an important characteristic of the predicted sets. For example, in extractive document summarization, the goal is to choose a set of sentences from a document that are not only high-quality but also diverse, so as to avoid redundancy in the summary. Kulesza & Taskar (2010) applied DPPs to structured objects, and used them to select multiple poses of people in an image, where the poses are "diverse" in the sense that they tend not to overlap.

However, DPPs conflate the modeling of two distinct characteristics: the *size* of the set, and its *content*. For example, a DPP would predict not only which sentences to include in a summary, but also how many. In some cases this can be beneficial; for instance, predicting the number of people in an image might be an important part of the pose identification task. However, for other applications the size of the desired set is often known in advance, or even adjusted on-the-fly at test time. For example, search systems frequently return a fixed number of results to the user, and that number might vary depending on the target platform.

In such situations, the effort spent by DPPs modeling the size of the set is wasteful. More importantly, because DPPs inextricably link size and content—essentially all degrees of freedom in the model affect both considerations—the need to consider set size can actually bias the modeling of content in a negative way. Finally, at test time there is no simple and justifiable way to control the size of predicted sets. This can be a serious limitation to practical use.

To address these problems, we introduce the k-DPP, which conditions a standard DPP on the event that the modeled set is of size k. This conditionalization, though simple in theory, necessitates new algorithms for model normalization and sampling. Naively, these tasks require exponential time, but we show that through an application of Newton's identities we can solve them exactly in time quadratic in k. Because k-DPPs focus all of their modeling capacity on content, they can be significantly more expressive than DPPs. Furthermore, because the underlying DPP can be re-conditionalized for different k, k-DPPs greatly simplify integration in applications that require test time control of set size.

Appearing in *Proceedings of the 28th International Conference on Machine Learning*, Bellevue, WA, USA, 2011.

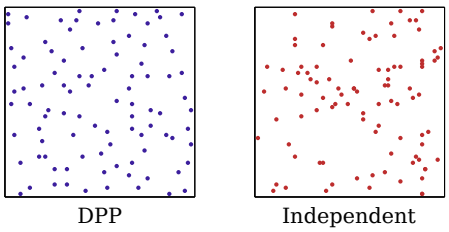

Figure 1. A set of points in the plane drawn from a DPP (left), and the same number of points sampled independently (right).

In Section 2 we give an overview of DPPs, and we introduce k-DPPs in Section 3. In Section 4 we show how to learn a mixture of k-DPPs from a labeled training set, and finally in Section 5 we apply our methods to a real-world image search problem. We show significant improvements in performance compared with Maximal Marginal Relevance (MMR), a popular technique for generating diverse result sets (Carbonell & Goldstein, 1998).

2. Determinantal point processes

A point process $\mathcal{P}$ on a discrete set $\mathcal{Y} = \{1, \ldots, N\}$ (for example, a collection of documents or images) is a probability measure on $2^{\mathcal{Y}}$, the set of all subsets of $\mathcal{Y}$. $\mathcal{P}$ is called a determinantal point process (DPP) if, when $\boldsymbol{Y}$ is a random set drawn according to $\mathcal{P}$, we have, for every $A \subseteq \mathcal{Y}$:

$$\mathcal{P}(A \subseteq \boldsymbol{Y}) = \det(K_A) \;, \tag{1}$$

for some positive semidefinite matrix $K \preceq I$ (all eigenvalues of K are less then or equal to 1) indexed by the elements of $\mathcal{Y}$. $K_A \equiv [K_{ij}]_{i,j \in A}$ denotes the restriction of K to the entries indexed by elements of A, and we adopt $\det(K_\emptyset) = 1$. We will refer to K as the marginal kernel, as it contains all the information needed to compute the probability of any subset A being included in $\boldsymbol{Y}$. A few simple observations follow from Equation (1):

$$\mathcal{P}(i \in \boldsymbol{Y}) = K_{ii} \tag{2}$$

$$\begin{aligned} \mathcal{P}(i, j \in \boldsymbol{Y}) &= K_{ii}K_{jj} - K_{ij}K_{ji} \\ &= \mathcal{P}(i \in \boldsymbol{Y})\mathcal{P}(j \in \boldsymbol{Y}) - K_{ij}^2. \end{aligned} \tag{3}$$

That is, the diagonal of K gives the marginal probabilities of inclusion for individual elements of $\mathcal{Y}$, and the off-diagonal elements determine the (anti-) correlations between pairs of elements: large values of K_{ij} imply that i and j tend not to co-occur. A DPP might therefore be used naturally to model diverse sets of items, for example in response to a search query. Note that DPPs cannot represent distributions where elements are *more* likely to co-occur than if they were independent; correlations are always negative.

Figure 1 shows the difference between sampling a set of points in the plane using a DPP (with K_{ij} inversely related to the distance between points i and j), which leads to a widely spread set with good coverage, and sampling points independently, where the points exhibit random clumping. Determinantal point processes, introduced to model fermions (Macchi, 1975), also arise in studies of non-intersecting random paths, random spanning trees, and eigenvalues of random matrices (Daley & Vere-Jones, 2003; Borodin & Soshnikov, 2003; Hough et al., 2006).

For the purposes of modeling real data, however, the most relevant construction of DPPs is not through K but via L-ensembles (Borodin, 2009). An **L-ensemble** defines a DPP via a positive semidefinite matrix L indexed by the elements of $\mathcal{Y}$:

$$\mathcal{P}_L(\boldsymbol{Y} = Y) = \frac{\det(L_Y)}{\det(L + I)} \;, \tag{4}$$

where I is the $N \times N$ identity matrix. As a shorthand, we will write $\mathcal{P}_L(Y)$ instead of $\mathcal{P}_L(\boldsymbol{Y} = Y)$ when the meaning is clear. Note that $\mathcal{P}_L$ is normalized due to the identity

$$\sum_{Y \subseteq \mathcal{Y}} \det(L_Y) = \det(L + I) \;. \tag{5}$$

K and L offer alternative representations of DPPs, and we can easily translate between the two; for example, we can compute the marginal kernel K for an L-ensemble:

$$K = (L + I)^{-1}L. \tag{6}$$

Note that K can be computed from an eigendecomposition of $L = \sum_{n=1}^{N} \lambda_n \boldsymbol{v}_n \boldsymbol{v}_n^\top$ by a simple rescaling of eigenvalues:

$$K = \sum_{n=1}^{N} \frac{\lambda_n}{\lambda_n + 1} \boldsymbol{v}_n \boldsymbol{v}_n^\top \;. \tag{7}$$

We can also similarly compute $L = K(I - K)^{-1}$, as long as the inverse exists.

Under both K and L representations, subsets that have higher diversity, as measured by the corresponding kernel, have higher likelihood. However, while K gives rise to marginal probabilities, L-ensembles directly model the probabilities of (exactly) observing each subset of $\mathcal{Y}$, which offers a convenient target for optimization. Furthermore, L need only be positive

Algorithm 1 Sampling from a DPP

Input: eigenvector/value pairs $\{(\boldsymbol{v}_n, \lambda_n)\}$
$J \leftarrow \emptyset$
for $n = 1, \ldots, N$ **do**
  $J \leftarrow J \cup \{n\}$ with prob. $\frac{\lambda_n}{\lambda_n+1}$
end for
$V \leftarrow \{\boldsymbol{v}_n\}_{n \in J}$
$Y \leftarrow \emptyset$
while $|V| > 0$ **do**
  Select y_i from $\mathcal{Y}$ with $\Pr(y_i) = \frac{1}{|V|} \sum_{\boldsymbol{v} \in V} (\boldsymbol{v}^\top \boldsymbol{e}_i)^2$
  $Y \leftarrow Y \cup y_i$
  $V \leftarrow V_\perp$, an orthonormal basis for the subspace of V orthogonal to $\boldsymbol{e}_i$
end while
Output: Y

semidefinite, while the eigenvalues of K are bounded above. For these reasons we focus our modeling efforts on DPPs represented as L-ensembles.

2.1. Inference

In addition to computing marginals (Equation (1)) and the normalizing constant (Equation (5)), a suprising number of other DPP inference operations are also efficient, despite the fact that we are modeling an exponential number of possible subsets Y.

For example, we can compute conditional probabilities:

$$\mathcal{P}(\boldsymbol{Y} = A \cup B \mid A \subseteq \boldsymbol{Y}) = \frac{\det(L_{A \cup B})}{\det(L + I_{\mathcal{Y} \setminus A})} , \qquad (8)$$

where $I_{\mathcal{Y} \setminus A}$ is the matrix with ones in the diagonal entries indexed by elements of $\mathcal{Y} \setminus A$ and zeros everywhere else. Conditional marginal probabilities $\mathcal{P}(B \subseteq \boldsymbol{Y} \mid A \subseteq \boldsymbol{Y})$ as well as inclusion/exclusion probabilities $\mathcal{P}(A \subseteq \boldsymbol{Y} \wedge B \cap \boldsymbol{Y} = \emptyset)$ can also be computed efficiently using eigen-decompositions of L and related matrices (Borodin, 2009).

Of particular interest here is the fact that we can efficiently sample from a DPP (Hough et al., 2006; Tao, 2009).

Theorem 1. *Let $L = \sum_{n=1}^N \lambda_n \boldsymbol{v}_n \boldsymbol{v}_n^\top$ be an orthonormal eigen-decomposition of a positive semidefinite matrix L, and let $\boldsymbol{e}_i$ be the ith standard basis N-vector (all zeros except for a 1 in the ith position). Then Algorithm 1 samples $\boldsymbol{Y} \sim \mathcal{P}_L$.*

Algorithm 1 offers some additional insights. Because the dimension of V is reduced by one on each iteration of the second loop, and because the initial dimension of V is simply the number of selected eigenvectors ($|J|$), the size of the subset $\boldsymbol{Y}$ is distributed as the number of successes in N Bernoulli trials where trial n succeeds with probability $\frac{\lambda_n}{\lambda_n+1}$. In particular, $|\boldsymbol{Y}|$ cannot be larger than $\operatorname{rank}(L)$, and we have:

$$\mathbb{E}[|\boldsymbol{Y}|] = \sum_{n=1}^N \frac{\lambda_n}{\lambda_n + 1} \qquad (9)$$

$$\operatorname{Var}(|\boldsymbol{Y}|) = \sum_{n=1}^N \frac{\lambda_n}{(\lambda_n + 1)^2} . \qquad (10)$$

While a full proof of Theorem 1 is beyond the scope of this paper, we will state a few of the important lemmas. First we need some terminology.

Definition 1. *A DPP is called* **elementary** *if every eigenvalue of its marginal kernel is in $\{0, 1\}$. We write $\mathcal{P}^V$, where V is a set of orthonormal vectors, to denote an elementary DPP with marginal kernel $K^V = \sum_{\boldsymbol{v} \in V} \boldsymbol{v}\boldsymbol{v}^\top$.*

Note that, due to Equation (7), elementary DPPs generally cannot be written as finite L-ensembles, since an eigenvalue of K is equal to 1 only if an eigenvalue of L is infinite.

Lemma 1. *If $\boldsymbol{Y}$ is drawn according to an elementary DPP $\mathcal{P}^V$, then $|\boldsymbol{Y}| = |V|$ with probability 1.*

Lemma 2. *An L-ensemble with kernel $L = \sum_{n=1}^N \lambda_n \boldsymbol{v}_n \boldsymbol{v}_n^\top$ is a mixture of elementary DPPs:*

$$\mathcal{P}_L = \frac{1}{\det(L + I)} \sum_{J \subseteq \{1, \ldots, N\}} \mathcal{P}^{V_J} \prod_{n \in J} \lambda_n , \qquad (11)$$

where V_J denotes the set $\{\boldsymbol{v}_n\}_{n \in J}$.

Lemma 2 says that the mixture weight of $\mathcal{P}^{V_J}$ is given by the product of the eigenvalues λ_n corresponding to the eigenvectors $\boldsymbol{v}_n \in V_J$, normalized by $\det(L + I) = \prod_{n=1}^N (\lambda_n + 1)$. This shows that the first loop of Algorithm 1 selects an elementary DPP $\mathcal{P}^V$ with probability equal to its mixture component. The remainder of the proof of Theorem 1 (omitted) shows that the second loop of the algorithm correctly samples $\mathcal{P}^V$. For full proofs of these lemmas and Theorem 1, see Tao (2009); Hough et al. (2006).

2.2. Size vs. content

As Algorithm 1 makes clear, a DPP models both the size of $\boldsymbol{Y}$, determined by the number of initially selected eigenvectors, and its content, determined by the span of those eigenvectors. In some instances this is a valuable property; for example, we may not know in advance how large $\boldsymbol{Y}$ should be, and we want the model to guess, as in multiple pose detection (Kulesza

& Taskar, 2010). However, for many applications, such as search, we want to select sets of fixed size, or to vary the size at test time. In these cases DPPs are modeling set size unnecessarily.

Furthermore, the burden of modeling set size actually interferes with the useful modeling of content. For example, an elementary DPP enforces a fixed set size k, but cannot even define a uniform distribution over the sets of size k (Tao, 2009). In the next section we introduce k-DPPs, which do not have this limitation.

3. k-DPPs

A k-DPP on a discrete set $\mathcal{Y}$ is a distribution over all subsets $Y \in \mathcal{Y}$ with cardinality k. In contrast to the standard DPP, which models both the size and content of $\boldsymbol{Y}$, the k-DPP models only content. In doing so, it sacrifices the ability to distinguish between sets of varying size, but gains additional flexibility in modeling the content of k-sets. In practice, this can be a worthwhile tradeoff.

A k-DPP is obtained simply by conditioning a standard DPP on the event that the set $\boldsymbol{Y}$ has cardinality k. Formally, a k-DPP $\mathcal{P}_L^k$ gives probabilities

$$P_L^k(Y) = \frac{\det(L_Y)}{\sum_{|Y'|=k} \det(L_{Y'})} , \tag{12}$$

where $|Y| = k$ and the parameter L is any positive semi-definite kernel.

While the normalizing constant for a DPP can be written in closed form, for k-DPPs the situation is somewhat more complicated. Naively, the sum in Equation (12) takes exponential time. However, it turns out the normalization is given by the kth elementary symmetric polynomial evaluated at λ. One way to see this is to examine the characteristic polynomial $\det(L - \lambda I)$ (see page 88, Gel'fand (1989)). We can also show it directly. Using Equation (5), we have:

$$\sum_{|Y'|=k} \det(L_{Y'}) = \det(L + I) \sum_{|Y'|=k} \mathcal{P}_L(Y') . \tag{13}$$

Applying Lemma 2,

$$\begin{aligned} &\det(L + I) \sum_{|Y'|=k} \mathcal{P}_L(Y') \\ &\quad = \sum_{|Y'|=k} \sum_{J \subseteq \{1,\ldots,N\}} \mathcal{P}^{V_J}(Y') \prod_{n \in J} \lambda_n \qquad (14) \\ &\quad = \sum_{|J|=k} \sum_{|Y'|=k} \mathcal{P}^{V_J}(Y') \prod_{n \in J} \lambda_n \qquad (15) \\ &\quad = \sum_{|J|=k} \prod_{n \in J} \lambda_n , \qquad (16) \end{aligned}$$

where we use Lemma 1 to conclude that $\mathcal{P}^{V_J}(Y') = 0$ unless $|J| = |Y'|$.

Define the kth power sum and kth elementary symmetric polynomial as follows:

$$p_k(\lambda_1, \ldots, \lambda_N) = \sum_{n=1}^{N} \lambda_n^k \tag{17}$$

$$e_k(\lambda_1, \ldots, \lambda_N) = \sum_{|J|=k} \prod_{n \in J} \lambda_n . \tag{18}$$

Then Equation (16) is just $e_k(\lambda_1, \ldots, \lambda_N)$. Newton's identities provide a recursion:

$$k e_k = e_{k-1} p_1 - e_{k-2} p_2 + e_{k-2} p_3 - \cdots \pm p_k . \tag{19}$$

We can compute the values of $p_1, \ldots, p_k$ in $O(Nk)$ time. The recursion for e_k is k levels deep, and each level takes $O(k)$ time. Thus we can efficiently normalize a k-DPP in $O(Nk + k^2)$ time, assuming an eigendecomposition of L is available.

3.1. Sampling from k-DPPs

Let e_k^N be a shorthand for $e_k(\lambda_1, \ldots, \lambda_N)$. Substituting Equation (16) back into Equation (12), we get

$$\mathcal{P}_L^k = \frac{1}{e_k^N} \det(L + I) \mathcal{P}_L , \tag{20}$$

and applying Lemma 2 and Lemma 1 yields

$$\mathcal{P}_L^k = \frac{1}{e_k^N} \sum_{|J|=k} \mathcal{P}^{V_J} \prod_{n \in J} \lambda_n . \tag{21}$$

Therefore a k-DPP is also a mixture of elementary DPPs, but it only gives nonzero weight to those of dimension k. Since the second loop of Algorithm 1 provides a means for sampling from any given elementary DPP, we can sample from a k-DPP if we can sample index sets J according to the corresponding mixture components. Again this is naively an exponential task, but we can do it efficiently using Newton's identities.

Let $\boldsymbol{J}$ be the desired random variable, so that $\Pr(\boldsymbol{J} = J) = \frac{1}{e_k^N} \prod_{n \in J} \lambda_n$ for $|J| = k$. The marginal probability of index N is given by

$$\Pr(N \in \boldsymbol{J}) = \frac{\lambda_N}{e_k^N} \sum_{\substack{J' \subseteq \{1,\ldots,N-1\} \\ |J'|=k-1}} \prod_{n \in J'} \lambda_n = \lambda_N \frac{e_{k-1}^{N-1}}{e_k^N} . \tag{22}$$

Furthermore, the conditional distribution for $\boldsymbol{J}$ given the decision to include or exclude N is of the same

Algorithm 2 Sampling from a k-DPP

Input: eigenvector/value pairs $\{(\boldsymbol{v}_n, \lambda_n)\}$, size k
$J \leftarrow \emptyset$
for $n = N, \ldots, 1$ **do**
  if $u \sim U[0,1] < \lambda_n \frac{e_{k-1}^{n-1}}{e_k^n}$ **then**
    $J \leftarrow J \cup \{n\}$
    $k \leftarrow k - 1$
    if $k = 0$ **then**
      break
    end if
  end if
end for
Proceed with the second loop of Algorithm 1
Output: Y

form:

$$\Pr(\boldsymbol{J} = \{N\} \cup J' | N \in \boldsymbol{J}) = \frac{1}{e_{k-1}^{N-1}} \prod_{n \in J'} \lambda_n \tag{23}$$

$$\Pr(\boldsymbol{J} = J | N \notin \boldsymbol{J}) = \frac{1}{e_k^{N-1}} \prod_{n \in J} \lambda_n \tag{24}$$

By induction, we get a simple algorithm for sampling from a k-DPP, given in Algorithm 2. The algorithm requires precomputing the values of $e_1^1, \ldots, e_k^N$, which in turn requires computing the corresponding power sums. Computing the power sums takes $O(Nk)$ time, and N Newton's identity recursions require an additional $O(Nk^2)$ time. Since the loop in Algorithm 2 executes at most N times and requires only a constant number of operations, sampling from a k-DPP requires $O(Nk^2)$ time overall, assuming an eigendecomposition of L. Furthermore, because the choice of k in the sampling algorithm is arbitrary, we can set it at test time, e.g., to sample sets of varying size as required by our application.

4. Learning

Because k-DPPs (and DPPs in general) give set probabilities as ratios of determinants (Equation (4)), likelihood is not convex in the kernel matrix L. This makes traditional likelihood-based learning objectives very unstable and difficult to optimize. Instead, we propose a simple method for learning a combination of k-DPPs that is convex and works well in practice.

Given a set $L_1, \ldots, L_D$ of available "expert" kernel matrices, we define the combination model

$$\mathcal{P}_\alpha^k = \sum_{d=1}^{D} \alpha_d \mathcal{P}_{L_d}^k \ , \tag{25}$$

where $\sum_d \alpha_d = 1$.

Because absolute human judgments of diversity tend to be extremely noisy, we assume that our labeled training data comprises comparative pairs $\{(Y_t^+, Y_t^-)\}_{t=1}^T$, where Y_t^+ is preferred over Y_t^-, $|Y_t^+| = |Y_t^-| = k$. We choose α to optimize a logistic loss measure:

$$\begin{aligned} \min_\alpha \quad & \mathcal{L}(\alpha) = \sum_{t=1}^{T} \log\left(1 + e^{\gamma\left[\mathcal{P}_\alpha^k(Y_t^-) - \mathcal{P}_\alpha^k(Y_t^+)\right]}\right) \\ \text{s.t.} \quad & \sum_{d=1}^{D} \alpha_d = 1 \ , \end{aligned} \tag{26}$$

where γ is a hyperparameter that controls how aggressively we penalize mistakes.

We optimize Equation (26) using projected gradient descent:

$$\nabla \mathcal{L} = \sum_{t=1}^{T} \frac{e^{\alpha^\top \delta^t}}{1 + e^{\alpha^\top \delta^t}} \delta^t \tag{27}$$

$$\delta_d^t = \gamma(\mathcal{P}_{L_d}^k(Y_t^-) - \mathcal{P}_{L_d}^k(Y_t^+)) \ . \tag{28}$$

Projection onto the simplex is achieved using standard algorithms (Bertsekas, 1999).

5. Experiments

We study the performance of k-DPPs on a real-world image search task, building on a wealth of work on creating diversity in search results, including the diversity heuristic Maximal Marginal Relevance (MMR) (Carbonell & Goldstein, 1998), diversity inducing metrics for simple probabilistic models of relevance (Chen & Karger, 2006), multi-armed bandit exploration/exploitation models (Radlinski et al., 2008), structured prediction models learned using structural SVMs (Yue & Joachims, 2008), and submodular optimization for blog search (El-Arini et al., 2009).

In order to evaluate our model, we define the task in a manner that allows us to directly evaluate it using inexpensive human supervision via Amazon Mechanical Turk. The goal is to choose, given two possible sets of image search results, the set that is more diverse. For comparison with MMR, which selects results iteratively and does not directly score entire sets, the two sets always differ by a single element. (Equivalently, we ask the algorithms to choose which of two images is a less redundant addition to a partial result set.) This setup defines a straightforward binary decision problem, and we measure performance using the zero-one loss. We use human judgments of diversity for our labeled training and testing data.

Table 1. Queries used for data collection.

CARS	CITIES	DOGS
chrysler	baltimore	beagle
ford	barcelona	bernese
honda	london	blue heeler
mercedes	los angeles	cocker spaniel
mitsubishi	miami	collie
nissan	new york city	great dane
porsche	paris	labrador
toyota	philadelphia	pomeranian
	san francisco	poodle
	shanghai	pug
	tokyo	schnauzer
	toronto	shih tzu

5.1. Data

By hand, we chose three image search categories and 8–12 queries for each category. (See Table 1.) For each query, we retrieved the top 64 results from Google Image Search, restricting the search to JPEG files that passed the strictest level of Safe Search filtering. Of those 64 results, we eliminated any that were no longer available for download. On average this left us with 63.0 images per query, with a range of 59–64.

We generated 960 instances for each category, spread evenly across the different queries. Each instance comprised a partial result set of five images plus two additional candidate images. The partial result sets were sampled using a k-DPP with a SIFT-based kernel (details below) to encourage diversity. The candidates were selected uniformly at random from the remaining images, except for 10% of the instances, which we reserved for measuring the performance of our human judges. For those instances, one of the candidates was identical to an image from the partial result set, making it the obviously more redundant choice.

We collected human diversity judgments using Amazon's Mechanical Turk. Annotators came from the general pool of Turk workers, and could label as many instances as they wished. Annotators were paid \$0.01 USD for each instance that they labeled. We presented all images at reduced scale; the larger dimension of each image was 250 pixels. The annotators were instructed to choose the candidate that they felt was less similar to the partial result set. We did not offer any specific guidance on how to judge similarity. Figure 2 shows a sample instance from each category.

Overall, workers chose the correct image for 80.8% of the calibration instances, suggesting only moderate levels of noise due to misunderstanding, inattention, etc. However, for non-calibration instances the task is inherently difficult and subjective. To keep noise in check, we had each instance labeled by five judges, and kept only those instances where four or more judges agreed. This left us with 408–482 labeled instances per category, or about half of the original instances.

5.2. Kernels

We built a set of 55 similarity kernels for the collected images. These kernels were used to define L-ensemble k-DPPs and to provide similarity measurements for MMR, discussed further below. Each kernel L^f is the Gram matrix of some feature function f; that is, $L^f_{ij} = f(i) \cdot f(j)$ for images i and j. We therefore specify the kernels through the feature functions used to generate them. All of our feature functions are normalized so that $\|f(i)\|_2^2 = 1$ for all i; this ensures that no image is *a priori* more likely than any other. Implicitly, we assume that all of the images in our set are equally *relevant* in order to isolate the modeling of *diversity*—our main focus in this work.

We use the following feature functions:

- COLOR (2 variants): Each pixel is assigned a coordinate in three-dimensional Lab color space. The colors are then sorted into axis-aligned bins, producing a histogram of either 8 or 64 dimensions.
- SIFT (2 variants): The images are processed with the `vlfeat` toolbox to obtain sets of 128-dimensional SIFT descriptors (Lowe, 1999; Vedaldi & Fulkerson, 2008). The descriptors for a given category are combined, subsampled to a set of 25,000, and then clustered using k-means into either 256 or 512 clusters. The feature vector for an image is the normalized histogram of the nearest clusters to the descriptors in the image.
- GIST: The images are processed using code from Oliva & Torralba (2006) to yield 960-dimensional GIST feature vectors characterizing properties like "openness," "roughness," "naturalness," etc.

In addition to these five feature functions, we include another five that are identical but consider only the center of the image, defined as the centered rectangle with dimensions half those of the original image. This gives ten basic kernels. We then create 45 pairwise combination kernels by concatenating every possible pair of the 10 basic feature vectors. This technique produces kernels that synthesize more than one source of information, offering greater flexibility.

Finally, we augment our kernels by adding a constant hyperparameter ρ to each entry. ρ acts a knob for

Figure 2. Sample labeling instances from each search category. The five images on the left form the partial result set, and the two candidates are shown on the right. The candidate receiving the majority of annotator votes has a blue border.

controlling the overall preference for diversity; as ρ increases, all images appear more similar, thus increasing repulsion. In our experiments, ρ was chosen independently for each method and each category to optimize performance on the training set.

5.3. Methods

We tested four different methods. For all of the methods, a training set consisting of 75% of the available labeled data (selected at random) was used to set hyperparameters, and evaluation was performed on the remaining 25%. In the following we use Y_t to denote the five-image partial result set for instance t, and $C_t = \{i_t^+, i_t^-\}$ to denote the set of two candidates, where i_t^+ is the candidate preferred by judges.

Best k-DPP:. Given a single kernel L, the k-DPP prediction is

$$k\text{DPP}_t = \arg\max_{i \in C_t} \mathcal{P}_L^6(Y_t \cup \{i\}) \ . \tag{29}$$

We select the kernel with the best zero-one accuracy on the training set, and apply it to the test set.

Mixture of k-DPPs:. We apply our learning method to the full set of 55 kernels. We map training instances to the form given in Section 4 as follows:

$$Y_t^+ = Y_t \cup \{i_t^+\} \tag{30}$$
$$Y_t^- = Y_t \cup \{i_t^-\} \ . \tag{31}$$

Optimizing Equation (26) on the training set yields a 55-dimensional mixture vector α, which is then used to make predictions on the test set:

$$k\text{DPPmix}_t = \arg\max_{i \in C_t} \sum_{d=1}^{55} \alpha_d \mathcal{P}_{L_d}^6(Y_t \cup \{i\}) \ . \tag{32}$$

Best MMR: Maximal Marginal Relevance is a standard technique for generating diverse sets of search results (Carbonell & Goldstein, 1998). The idea is to build a set iteratively by adding on each round a result that maximizes a weighted combination of relevance (with respect to the query) and diversity, measured as the maximum similarity to any of the previously selected results. For our experiments, we assume relevance is uniform; hence we merely need to decide which of the two candidates has the smaller maximum similarity to the partial result set. For a given kernel L, the MMR prediction is

$$\text{MMR}_t = \arg\min_{i \in C_t} \left[\max_{j \in Y_t} L_{ij} \right] \ . \tag{33}$$

As with k-DPPs, we select the single best kernel on the training set, and apply it to the test set.

Mixture MMR: We also test MMR using a mixture of similarity kernels. We use the same training approach as for k-DPPs, but replace the probability score $P_\alpha^k(Y_t \cup \{i\})$ with the negative cost

$$-c_\alpha(Y_t, i) = -\max_{j \in Y_t} \sum_{d=1}^{D} \alpha_d (L_d)_{ij} \ . \tag{34}$$

Significantly, this substitution makes the optimization non-smooth and non-convex, unlike the k-DPP optimization. In practice this means the global optimum is

Table 2. Percentage of real-world image search examples judged the same way as the majority of human annotators. Bold results are significantly higher than others in the same row with 99% confidence.

Cat.	Best MMR	Best k-DPP	Mixture MMR	Mixture k-DPP
cars	55.95	57.98	59.59	**64.58**
cities	56.48	56.31	60.99	61.29
dogs	56.23	57.70	57.39	**59.84**

Table 3. Kernels receiving the highest average weights for each category (shown in parentheses). Ampersands indicate kernels generated from pairs of feature functions.

cars	color-8-center & sift-256	(0.13)
	color-8-center & sift-512	(0.11)
	color-8-center	(0.07)
cities	sift-512-center	(0.85)
	gist	(0.08)
	color-8-center & gist	(0.03)
dogs	color-8-center	(0.39)
	color-8-center & sift-512	(0.21)
	color-8-center & sift-256	(0.20)

not easily found, and local optima may perform inconsistently. In our experiments we use the local optimum found by projected gradient descent.

5.4. Results

Table 2 shows the mean zero-one accuracy of each method for each query category, averaged over 100 random train/test splits. Statistical significance was computed by bootstrapping. With and without learning a mixture, k-DPPs outperform MMR on two of the three categories, significant at 99% confidence.

Table 3 shows, for the k-DPP mixture model, the kernels receiving the highest weights for each search category (on average over 100 train/test splits). Combined-feature kernels appear to be useful, the three categories exhibit significant differences in what annotators deem diverse.

6. Conclusion

We introduced k-DPPs, conditionalized DPPs that directly model sets of fixed size, showing how to efficiently normalize and sample from them. We found that k-DPPs significantly outperformed MMR for identifying diverse image search results. Future work includes the study of alternative learning formulations that directly optimize the kernel of a k-DPP.

References

Bertsekas, D. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1999.

Borodin, A. and Soshnikov, A. Janossy densities. I. Determinantal ensembles. *Journal of Statistical Physics*, 113 (3):595–610, 2003.

Borodin, Alexei. Determinantal point processes, 2009. URL http://arxiv.org/abs/0911.1153.

Carbonell, J. and Goldstein, J. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proc. SIGIR*, 1998.

Chen, H. and Karger, D.R. Less is more: probabilistic models for retrieving fewer relevant documents. In *Proce. SIGIR*, pp. 429–436, 2006.

Daley, D.J. and Vere-Jones, D. *An introduction to the theory of point processes: volume I: elementary theory and methods*. Springer, 2003.

El-Arini, Khalid, Veda, Gaurav, Shahaf, Dafna, and Guestrin, Carlos. Turning down the noise in the blogosphere. In *Proc. KDD*, 2009.

Gel'fand, I.M. *Lectures on linear algebra*. Dover, 1989. ISBN 0486660826.

Hough, J.B., Krishnapur, M., Peres, Y., and Virág, B. Determinantal processes and independence. *Probability Surveys*, 3:206–229, 2006.

Kulesza, Alex and Taskar, Ben. Structured determinantal point processes. In *Proc. Neural Information Processing Systems*, 2010.

Lowe, D.G. Object recognition from local scale-invariant features. In *Proc. ICCV*, 1999.

Macchi, O. The coincidence approach to stochastic point processes. *Advances in Applied Probability*, 7(1):83–122, 1975.

Oliva, A. and Torralba, A. Building the gist of a scene: The role of global image features in recognition. *Progress in brain research*, 155:23–36, 2006. ISSN 0079-6123.

Radlinski, F., Kleinberg, R., and Joachims, T. Learning diverse rankings with multi-armed bandits. In *Proc. ICML*, 2008.

Tao, Terence. Determinantal processes. http://terrytao.wordpress.com/2009/08/23/determinantal-processes/, August 2009.

Vedaldi, A. and Fulkerson, B. VLFeat: An open and portable library of computer vision algorithms. http://www.vlfeat.org/, 2008.

Yue, Y. and Joachims, T. Predicting diverse subsets using structural SVMs. In *Proc. ICML*, 2008.