Practical Diversified Recommendations on YouTube with Determinantal Point Processes

Mark Wilhelm, Ajith Ramanathan, Alexander Bonomo, Sagar Jain, Ed H. Chi, Jennifer Gillenwater
Google Inc.
{wilhelm,ajith,bonomo,sagarj,edchi,jengi}@google.com

ABSTRACT

Many recommendation systems produce result sets with large numbers of highly similar items. Diversifying these results is often accomplished with heuristics, which are impoverished models of users' desire for diversity. However, integrating more complex statistical models of diversity into large-scale, mature systems is challenging. Without a good match between the model's definition of diversity and users' perception of diversity, the model can easily degrade users' perception of the recommendations. In this work we present a statistical model of diversity based on determinantal point processes (DPPs). We train this model from examples of user preferences with a simple procedure that can be integrated into large and complex production systems relatively easily. We use an approximate inference algorithm to serve the model at scale, and empirical results on live YouTube homepage traffic show that this model, coupled with a re-ranking algorithm, yields substantial short- and long-term increases in user satisfaction.

ACM Reference Format:
Mark Wilhelm, Ajith Ramanathan, Alexander Bonomo, Sagar Jain, Ed H. Chi, Jennifer Gillenwater. 2018. Practical Diversified Recommendations on YouTube with Determinantal Point Processes. In *Proceedings of International Conference on Information and Knowledge Management (CIKM'18).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Online recommendation services often present content in the form of a feed—an ordered list of items through which the user browses. Examples include the YouTube mobile homepage feed and the Facebook news feed. The goal is to select and order a set of k items such that the utility of the set is maximized. Often times recommenders do this by ranking based on item quality—assigning each item i a pointwise quality score, q_i, and sorting by this score. However, this is sub-optimal as the pointwise estimator ignores correlations between the items. For example, given that a basketball video has already been shown on the page, it may now be less useful to show another basketball video. This is exacerbated by the fact that similar videos tend to have similar quality scores. Unfortunately, even if we build a good set-wise estimator, scoring every possible permutation of the ranked list is prohibitively expensive.

In this paper, we apply a particular machine learning model called a determinantal point process (DPP) [4, 13, 22], which is a probabilistic model of repulsion that can be used to diversify sets of recommended items (e.g., lists of videos, books, or search results) [7, 14, 20, 21]. One key aspect of a DPP is that it can efficiently score an entire list of items rather than scoring each item individually, allowing us to better take into account item correlations.

Implementing a DPP-based solution in a mature recommendation system is non-trivial. First, the training methods for DPPs are significantly different from those used in typical recommender systems [3, 12, 14, 20, 21, 26, 27]. Second, integrating the DPP optimization with existing recommenders is complex. One option would be to retool the entire infrastructure in terms of set-wise recommendations, but that would discard the large investment in, and the sophistication of, the existing pointwise estimators. Instead, we use DPPs on top of existing infrastructure as a last-layer model. This allows the various underlying system components to evolve independently. More specifically, for a large-scale recommendation system, we build a DPP using two inputs: 1) pointwise estimators from a deep neural network built for recommendations [9], which gives us a high-precision estimate of item quality q_i, and 2) pairwise item distances D_{ij} computed in a sparse semantic embedding space. (e.g., [19]). From these inputs, we construct a DPP and apply it to the top n items in a feed. Our approach has the advantage of enabling teams of researchers to continue to develop the q_i and D_{ij} estimators simultaneously with our development of a set-wise scoring system. We can therefore achieve our diversification goals while leveraging existing investments in a large-scale prediction system. Empirical results on YouTube show substantial short- and long-term increases in user satisfaction.

Our contributions are:

(1) We offer a simple and effective procedure for set-wise recommendations by leveraging DPPs. We define a parameterization and learning algorithm for DPPs that makes use of pointwise quality scores for items and pairwise distances between items.
(2) We describe a practical and modular approach which can be applied in the context of latency-sensitive, large-scale recommender systems.
(3) We offer both offline and online empirical results verifying that our approach improves recommendation accuracy on top of a mature, large-scale recommender system.

The paper is organized as follows. We start with related works in §2. We describe the diversification needs in the current recommendation system in §3, defining basic terminology in §3.2. In §4, we briefly review DPPs, then describe our current choice of DPP kernel, work-in-progress on a more complex kernel, and a ranking algorithm that makes use of these kernels. Finally, we provide a summary of our online experimental results in §5 and end by offering a few concluding remarks in §6.

CIKM'18, October 2018, Turin, Italy

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

2 RELATED WORK

Current recommender research is generally focused on improving the pointwise estimate q_i—a prediction of how much a user will enjoy one particular item. This line of research was initially started over 20 years ago with user-based collaborative filtering [34] and item-based collaborative filtering [36], and then refined using matrix factorization techniques [19]. In our system, we now obtain these pointwise estimates from deep neural networks, in which a user's preference features are combined with the the item features to estimate how much the user will enjoy that content [9].

During the course of these refinements, there was also significant study of users' need for novelty and diversity in the recommendation results [16, 24, 29, 39, 41, 43, 45]. Similarly, there has been significant work on diversification in information retrieval systems such as web search [6, 8, 10, 11, 15, 33, 35, 40, 42]. Considering all of this literature, researchers have proposed many notions of diversification. Here we briefly summarize and contrast two different perspectives on the purpose of content diversification.

2.1 Diversification to Facilitate Exploration

First, diversification is sometimes seen as a way to facilitate exploration; showing the user more diverse content will (A) help them discover new topics of interest or (B) help the recommender system discover more about the user.

For discovering user intent, there is a thread of work in information retrieval on using taxonomy to resolve ambiguity in user intent [2, 35]. For instance, IA-Select in [2] uses a taxonomy to cover an ambiguous query, and then aims to maximize the probability that the user will select at least one returned result. Santos et al. [35] estimate how well a ranked result covers an uncovered aspect of the answer for an ambiguous query. Whereas these methods require a problem-specific taxonomy, the solution we present only requires uncalibrated item distances (the calibration is learned as part of the training procedure).

For facilitating topic discovery, if a topic contains multiple aspects, then one can further divide the topic into subtopics, and then make sure that each subtopic is well-covered by the results retrieved [10, 40, 42]. For instance, Dang et al. [10] proposes to return a result list that has per-topic coverage that is proportional to that topic's popularity. As another example, Perez et al. [32] uses categories of businesses to ensure recommendation results for a local business recommendation problem has sufficient topical coverage. In [23], Kwon and Adomavicius argue that users essentially want a multi-criteria rating system, in which they can specify which aspects of the recommendation they want. In contrast to these methods, we are able to learn the appropriate amount of coverage based directly on user behavior.

Perhaps it is worth noting that while exploration likely does happen to some extent in all recommenders, imperfect information about user preferences and correlated recommendations are fundamentally orthogonal problems. Exploration is still needed in the presence of uncorrelated recommendations, and diversification is still needed in the presence of perfect information.

Somewhat consistent with the exploration perspective on diversity is that it is a secondary product objective. This perspective suggests a fundamental trade-off between diversity and utility, and can be seen in work that focuses on increasing a diversity metric as much as possible, without hurting the utility too much. In recent work that is similar to ours, Chen et al. [7] described the use of DPPs to optimize exploration without hurting the user utility. Their DPP kernel parameterization is different, and our work offers not just offline experiments but also a large-scale online experiment. More importantly, in contrast, we optimize for user utility while increasing diversity using DPP.

2.2 Diversification in Service of Utility

A different perspective on diversity, and the one we adopt for this work, is that diversity operates directly in service of utility—by appropriately diversifying impressions, one can maximize the feed's utility. From this perspective diversity is purely about the correlation of interactions, and increasing diversity means replacing *redundant* video impressions with alternatives that a user is more likely to concurrently enjoy. These new videos generally have lower individual scores but lead to a better page overall.

Concisely, one way of achieving diversity is avoiding redundancy, which is particularly important for recommender systems [5, 30, 32, 43, 45]. For instance, in their seminal work in 2005, Ziegler et al. [45] minimize the similarity between recommended items using a greedy algorithm with a taxonomy of books. The output is then merged with a non-diversified result list using a diversification factor. In another seminal work in information retrieval, Carbonell and Goldstein [5] propose the maximal marginal relevance (MMR) method. This method involves iteratively selecting one item at a time. The score of an item under consideration is proportional to its relevance minus a penalty term that measures its similarity to previously selected items. Other explicit notions of redundancy are studied in [32], which uses a decay function on pairwise similarities. More recently, Nassif et al. [30] describe an approach using submodular optimization to diversify music recommendation. Lin and Bilmes [25] describe a way to use submodular functions to perform document summarization, a task with similar coverage goals as set diversification tasks. Tschiatschek et al. [38] describe an approach using submodular maximization to select sequences of items, while Teo et al. [37] describe using submodular diversification to re-rank top items based on category. Our goals are quite similar in nature, but use a different optimization technique. Additionally we do not take item diversity as an a priori goal; our aim is simply trying to increase the number of positive user interactions by making diversity information available to the overall recommendation system. One can imagine iterating on the model presented here to express a personalized notion of diversity. The recommended content feed is also a convenient context for this approach, since (unlike in search) users are typically not looking for a specific item and may interact with multiple items in the course of a session.

The notion of redundancy can be further broken up into two separate relevance notions: substitutes and complements. These notions have been employed by several recommender systems [28, 44]. In e-commerce recommendation applications, before the user makes a purchasing decision, it might be more helpful to offer substitutes of candidates under consideration, while complement products might be offered after the user has made a purchase.

2.3 Related Works Summary & Design Choices

In summary, many researchers before us have studied how to improve diversity in both recommendation and search results. Some researchers deal with several of these diversity notions at the same time. For instance, Vargas et al. [39] addresses coverage and redundancy, as well as the size of the recommendation list. We are interested in a technique that works well in practice in a large-scale recommendation system that can be served to hundreds of millions of users per day. The notion of diversity should be flexible enough that it can evolve over time. As a result, we chose not to pursue taxonomic or topic-coverage approaches, as they require some explicit representation of diversity (e.g., an explicit guess at the user's intent or topic coverage).

Instead, we propose an approach using *determinantal point processes* (DPPs) [4, 7, 13, 22]. DPP is a set-wise recommendation model that only requires two explicit and natural elements: how good is each item for the user, and how similar are each pair of items. As a result, our focus is on eliminating redundancy.

3 BACKGROUND

3.1 YouTube Homepage Feed Overview and the Need for Diversification

The overall structure of the system for generating the video recommendations on a user's YouTube mobile homepage feed is illustrated in Figure 1. The system is comprised of three phases: (1) candidate generation, wherein the feed items are selected from a large catalogue, (2) ranking, which orders the feed items, and (3) policy, which enforces business needs such as requiring that some content appear at a specific position on the page. Phases (1) and (2) both make heavy use of deep neural networks [9].

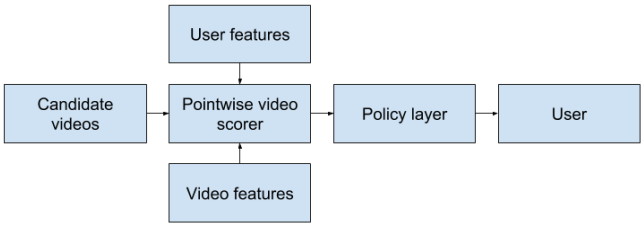

Figure 1: The basic serving scheme.

Candidate generation is substantially influenced by the previous behavior of the user on our system, and computes relatively simple measures of how well items match user preferences. For example, co-utility is one measure that is used: if a user enjoyed video A, and many other users who enjoyed A also enjoyed B, then B might be selected in the candidate generation phase. The ranking phase also makes heavy use of user features, but additionally relies on richer item features (such as embeddings of the video in some semantic space). As one might expect, the ranking phase tends to give similar videos similar utility predictions, leading to feeds that have repetitive content and, often, runs of very similar videos.

In order to mitigate the redundancy problem, at first, we introduced heuristics in the spirit of [32, 45] to the policy layer, such as a requirement that an individual uploader can contribute no more than n items to any user's feed. While this rule is somewhat effective, our experience is that it interacts quite poorly with the underlying recommendation system. Since the candidate generation and ranking layers are unaware of this heuristic, they make suboptimal predictions by wasting space on items that will never be presented. Furthermore, as the first two layers evolve over time, we need to repeatedly retune the parameters of the heuristics—a task that is expensive and hence in practice is not done with enough frequency to maintain much of the rule's effectiveness. Finally, the interactions between multiple types of heuristics yields, in practice, a recommendation algorithm that is very hard to understand. The result is a system that is suboptimal and difficult to evolve.

3.2 Definitions

To be more precise, let us denote the observed interactions of a user with items in a given feed as a binary vector y, (e.g., $y = [0, 1, 0, 1, 1, 0, 0, \ldots]$), where it is understood that the user typically will not look at the entire feed, but will start at the lower numbered indices. Our present goal is to maximize the total number of interactions:

$$G' = \sum_{u \sim \text{Users}} \sum_{i \sim \text{Items}} y_{ui} \,. \tag{1}$$

In order to train models from records of previous interactions, we try to select the parameters of the model to maximize the cumulative gain by reranking the feed items:

$$G = \sum_{u \sim \text{Users}} \sum_{i \sim \text{Items}} \frac{y_{ui}}{j} \,, \tag{2}$$

where j is the new rank that the model assigns to an item. This quantity increases as we rank interactions more highly. (In practice, we minimize jy_{ui} instead of maximizing $\frac{y_{ui}}{j}$, but the two expressions have the same optima.) In the following discussion, we will drop the u subscript for simplicity, although all values should be assumed to differ on a per-user basis

Let us further assume we are provided with some black box estimates of y's quality:

$$q_i \approx \mathrm{P}(y_i = 1 \mid \text{features of item } i) \,. \tag{3}$$

The obvious ranking policy is to sort the items according to q. Note though that q_i is a function of only a single item. If there are many similar items with similar values of q_i they will be ranked adjacent to each other, which may lead to the user abandoning the feed. Given that our ultimate goal is to maximize the feed's total utility, we call two items similar when:

$$\mathrm{P}(y_i = 1, y_j = 1) < \mathrm{P}(y_i = 1)\mathrm{P}(y_j = 1) \,. \tag{4}$$

In other words, they are negatively correlated when presented together—suggesting one of them is redundant. If there are similar items in the feed, then sorting by q is no longer the optimal policy.

Let us further assume we are provided with black box item distances:

$$D_{ij} = \text{distance}(\text{item } i, \text{item } j) \in [0, \infty) \,. \tag{5}$$

These distances are assumed to be 'uncalibrated', in the sense that they are not required to be directly related to Equation 4. For example, if the items in question are newspaper articles, D could be a Jaccard distance of the tokenized words in each article. The goal now is to produce a ranking policy based on q, D, and y which achieves a smaller value of G than simply sorting by q. Ideally

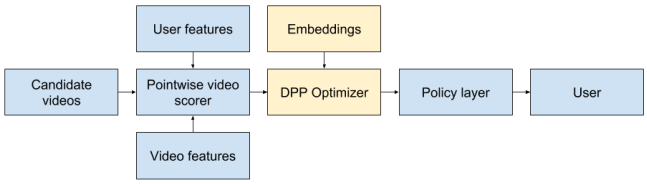

Figure 2: The new serving scheme.

this can be done in a way which integrates and evolves well with existing infrastructure.

3.3 Design Desiderata

If item similarity (as defined in Equation 4) exists in the dataset, and the dataset is sufficiently large, then our goal can likely be achieved by a wide variety of different methods. We favor a method which:

(1) fits well into the existing logical framework of building machine-learned estimators of observable physical events,
(2) can gracefully scale in complexity over time, and
(3) can be applied without requiring huge changes to existing systems and expertise.

Heuristics can be effective [45] but are not ideal. For example, imagine enforcing the rule that within a window of n adjacent items, no two items may have $D_{ij} < \tau$. A number of issues arise:

(1) This rule operates independently of q, meaning that it will demote high-scoring items under the same conditions as it does low-scoring items. Independent improvements to the accuracy of q may be lost after applying the policy.
(2) Parameters n and τ can be found by brute force grid search, but adding complexity will become prohibitive, as training time will be exponential in the number of parameters.
(3) It is not entirely obvious how to extend the rule to make incremental improvements over time, beyond somehow incorporating q.
(4) It cannot be used as a generative model to create synthetic datasets for offline validation.

An important point is that this heuristic implicitly treats the redundancy problem as a fundamentally different objective from maximizing utility. In fact, it suggests the hypothesis that improving diversity might reduce utility (at least in the short term), since it throws away items that have high q values. In contrast, our proposed approach considers utility on *pairs* of items (via the anti-correlation described in Equation 4), and hence is able to use utility itself to better justify demoting certain items.

Of course, it is possible to define a heuristic based upon anti-correlation, such as "no two items are allowed in the same feed if $\frac{P(y_i=1, y_j=1)}{P(y_i=1)P(y_j=1)}$ is below x". However, as mentioned above, this rule does not account for q, would require frequent re-tuning of the parameter x, and even with regular tuning is not flexible enough to capture exactly the behavior that we desire. Hence, in place of such heuristic rules, we introduce DPPs into the system as a more principled way of diversifying recommendations.

We insert the DPPs before the policy layer, but after the pointwise scoring layer (see Figure 2). This allows us to leverage the investment in an extremely sophisticated pointwise scorer, and also ensures that business policies are respected.

4 METHOD

4.1 DPP Overview

We start with a high-level overview of determinantal point processes (DPPs). A point process $\mathcal{P}$ on a set $\mathbb{S} = \{1, 2, \ldots, N\}$ (e.g., a set of N videos in a user's YouTube mobile homepage feed) is a probability distribution on the powerset of $\mathbb{S}$ (the set of all subsets of $\mathbb{S}$). That is, $\forall S \subseteq \mathbb{S}$, $\mathcal{P}$ assigns some probability $\mathcal{P}(S)$, and $\sum_{S \subseteq \mathbb{S}} \mathcal{P}(S) = 1$. DPPs represent a family of probability distributions whose parameters can be tuned such that the probability of a subset, $\mathcal{P}(S)$, is proportional to a combined measure of the quality of the items in S and the diversity of these items. Thus, finding the set $\max_{S:|S|=k} \mathcal{P}(S)$ is one way of selecting a high-quality and diverse subset of k items from a larger pool of N items.

As mentioned in Section 2, there are many reasonable measures that take into account both item quality and diversity, such as the maximal marginal relevance (MMR) approach [5]. The advantage of using DPPs is two-fold: 1) DPPs can out-perform measures such as MMR on recommendation tasks [20], and 2) a DPP is a probabilistic model. This latter point means that we can take advantage of algorithms for probabilistic operations like marginalization, conditioning, and sampling. The availability of these operations aligns well with our goal of building a system that can gracefully scale in complexity over time.

We now describe how we use a DPP to model user behavior. Recall that for a feed with N items, the length-N binary vector y indicates which videos from the feed the user interacted with. Let Y denote the index set of these items (e.g., for $y = [0, 1, 0, 0, 1, 1]$ we have $Y = \{2, 5, 6\}$). Then we assume that a user u's behavior is modeled by a DPP with probability distribution $\mathcal{P}$ in the following manner: $Y \sim \mathcal{P}_u$. That is, the set of videos interacted with, Y, represents a draw from the probability distribution defined by a user-specific DPP.

Even though a DPP defines a probability distribution over an exponential number of sets (all 2^N subsets of $\mathbb{S} = \{1, 2, \ldots, N\}$), it can be compactly parameterized by a single $N \times N$ positive semi-definite kernel matrix [4], which we will call L. More concretely, the probabilities of a DPP can be written as determinants of submatrices of L:

$$\mathcal{P}(Y) = \frac{\det(L_Y)}{\sum_{Y' \subseteq \mathbb{S}} \det(L_{Y'})} \,. \tag{6}$$

where L_Y is L restricted to only those rows and columns which are indexed by Y (e.g., for $Y = \{2, 5, 6\}$, the matrix L_Y is size 3×3). Note that the denominator in Equation 6 is simply a normalizing term, and, due to various properties of determinants, it can actually be written and computed efficiently as a single determinant:

$$\sum_{Y \subseteq \mathbb{S}} \det(L_Y) = \det(L + I) \,, \tag{7}$$

where I is the identity matrix.

To see how $\det(L_Y)$ can define a balanced measure of the quality and diversity of a set of items, it helps to think of the entries of L in the following manner: 1) a diagonal entry L_{ii} is a measurement of the quality of an item i, and 2) an off-diagonal element L_{ij} is a scaled measurement of the similarity between items i and j. With these intuitions, let's consider a case where $|Y| = 2$. If $Y = \{1, 2\}$,

then:

$$L_Y = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix} . \quad (8)$$

The determinant of this submatrix is: $\det(L_Y) = L_{11}L_{22} - L_{12}L_{21}$. So, it is a product of item qualities minus scaled item similarities. The expression for determinant is more complex for larger submatrices, but a similar intuition holds there as well.

In the following sections, we discuss various ways in which L can be constructed from the readily available system inputs, such as the pointwise item quality scores, q, that Section 3.2 described.

4.2 Kernel Parameterization

In our current deployment, as shown in Figure 2, diversification happens fairly late in the pipeline, so a typical input set size is $N = 100$. For each of these N videos, we have two main input features: a personalized quality score q, and a sparse embedding ϕ which is derived from the topical content of the video. These features are generated by completely independent subsystems. By stacking our diversification system on top of them we take advantage of the continuous improvements of these subsystems.

For the initial introduction of DPPs into our system, we first used a relatively simple parameterization of the $N \times N$ DPP kernel matrix L:

$$L_{ii} = q_i^2 \quad (9)$$

$$L_{ij} = \alpha q_i q_j \exp\left(-\frac{D_{ij}}{2\sigma^2}\right) , \text{ for } i \neq j . \quad (10)$$

Each D_{ij} is a computed from ϕ_i and ϕ_j; Section 5 describes the exact embedding ϕ and distance function that we found worked best in practice. The α and σ are free variables. Note that this formulation is identical to a standard (Gaussian) radial basis function (RBF) kernel when $\alpha = 1$. For smaller values, $\alpha \in [0, 1)$, the off-diagonal of the matrix is scaled down, which essentially corresponds to considering all items to be more diverse. For larger values, $\alpha > 1$, the off-diagonal of the matrix is scaled up, with the opposite effect of making all items more similar. As α grows, the probability of small sets grows while the probability of large sets shrinks. Thus, a large α is a good fit for user behavior in the setting where a relatively small subset of the videos in a feed are interacted with ($|Y|$ is small).

It is valuable for us to be able to use large α, because, as we will see in Section 4.3, it provides a better fit to real user data. However, there is one technical issue with allowing $\alpha > 1$. Recall from the Equation 6 that to have a proper DPP, the kernel matrix L must be positive semi-definite (PSD). The PSD condition ensures that the determinants of all submatrices of L are non-negative. This is important because the probability of a set Y is proportional to $\det(L_Y)$, and negative "probabilities" do not make sense. If we allow $\alpha > 1$ though, this has the potential to make L non-PSD. In practice, we resolve this issue by simply projecting any non-PSD matrix that a large α value produces back to the space of PSD matrices. (The projection is simple: we compute L's eigendecomposition and replace any negative eigenvalues with zeros.)

4.3 Training Approach

Our training set consists of approximately forty thousand examples sampled from one day of data collected from the YouTube mobile homepage feed. Each training example is a single homepage feed impression: a single instance of a user going to the YouTube mobile homepage and being presented with an ordered list of recommended videos. For each such impression we have a record of which videos a user enjoyed, which constitutes the set Y. We note that there is a partial-label bias associated with training the model from such data, since we only observe users' interactions with the videos that we chose to present them with in the past, rather than interactions with videos chosen uniformly at random. Generally, we use the same style of solutions to this issue as have been used in the past when training pointwise models, such as using an ϵ-greedy exploration strategy.

For the basic kernel described in the previous section there are only two parameters, α and σ, so we can simply do a grid search to find the values that maximize the cumulative gain (Equation 2). Figure 3 shows the cumulative gain obtained for various choices of α and σ. The darker the color, the worse the result. Interestingly, one can observe the catastrophic cliff in the upper right quadrant, and the subsequent plateau. This has to do with the DPP kernels for training examples becoming increasingly non-PSD. Recall that as α grows the off-diagonal of L grows, increasing the chance of a non-PSD L. Since the off-diagonal also grows somewhat with σ, large α, σ combinations result in non-PSD matrices for many of the training examples. Intuitively then, it might seem like the entire upper right corner of the plot should have low cumulative gain values, rather than the low values being concentrated in the observed band. However, also recall that we project any non-PSD matrices back to the PSD space. This projection is not linear with respect to α or σ and so the quality of the matrices after projection cannot be expected to necessarily correlate with our intuition about those parameters. Overall, we find that the highest cumulative gain is achieved in the mid-range for σ and in the upper half of the range for α. The L kernels produced by these parameters are mostly PSD, so only an occasional training example's kernel requires projection.

4.4 Deep Gramian Kernels

As discussed earlier, one of the main advantages of using DPPs over heuristics is that DPPs allow us to build a system that scales gracefully in complexity over time. We argued that the complexity of a heuristic scales poorly, because to tune it we have to do a grid search over its parameters, and so the runtime for training a heuristic is exponential in the number of parameters. In this section, we discuss how, with DPPs, we can move beyond grid searches to efficiently train a model with many parameters.

There is a substantial body of work on learning DPP kernel matrices that are parameterized in a variety of ways [3, 12, 14, 20, 21, 26, 27]. Such work usually seeks to maximize the log-likelihood of the training data. More concretely, suppose that:

- the parameters of L are some length-r vector $\boldsymbol{w}$, and
- we have M training examples, each consisting of: 1) a set of N items, and 2) the subset Y of these items that a user interacted with.

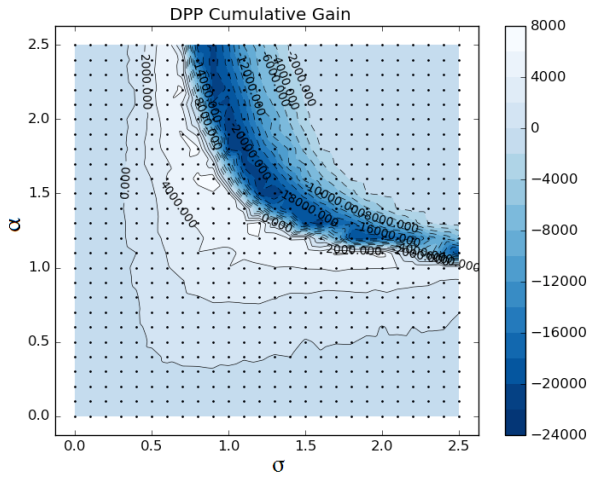

Figure 3: Cumulative gain for a grid of α and σ values on a dataset from the YouTube mobile homepage feed.

Let $L(\boldsymbol{w})$ be the $N \times N$ kernel matrix induced by the parameters $\boldsymbol{w}$. Then the log-likelihood of the training data is:

$$\text{LogLike}(\boldsymbol{w}) = \sum_{j=1}^{M} \log(\mathcal{P}_{L(\boldsymbol{w})}(Y_j)) \tag{11}$$

$$= \sum_{j=1}^{M} \left[\log(\det(L(\boldsymbol{w})_{Y_j})) - \log(\det(L(\boldsymbol{w}) + I)) \right] , \tag{12}$$

where Y_j is the subset of items from training example j that the user interacted with. The ability to use log-likelihood as an objective function allows us to learn DPP parameters with more sophisticated (and more efficient) methods than grid search.

We have begun to explore learning a kernel with many more parameters than the α and σ of the previous section, by using gradient descent on LogLike. We still use as input the ϕ embeddings that characterize video content. For the personalized video quality scores though, rather than a scalar score q_i, we are able to get from existing infrastructure an entire vector of quality scores $\boldsymbol{q}_i$, so we use this vector to make our model more general. (Each entry of the vector $\boldsymbol{q}_i$ captures some aspect of what might make a video a good choice for a user.) The full kernel $L(\phi, \boldsymbol{q})$ that we learn from this input data can be expressed in the following manner:

$$L_{ij} = f(\boldsymbol{q}_i)\boldsymbol{g}(\phi_i)^T \boldsymbol{g}(\phi_j) f(\boldsymbol{q}_j) + \delta \mathbb{1}_{i=j} , \tag{13}$$

where f and $\boldsymbol{g}$ are separate stacks in a neural network. (δ is simply a regularization parameter that we have for now fixed at a small value.) Note that the quantity $f(q_i)$ is a scalar, while $\boldsymbol{g}(\phi_i)$ is a vector. The neural network for computing f is relatively shallow, while $\boldsymbol{g}$'s network is deeper, and effectively re-embeds ϕ in a space which better describes utility correlation of videos (see Figure 4). We also note that, unlike the basic kernel parameterization discussed earlier, where large values of α could result in non-PSD L, this more complex parameterization is actually guaranteed to always produce PSD matrices without need for projection. This follows from the fact that this particular construction of L makes it a Gramian matrix, and all such matrices are PSD.

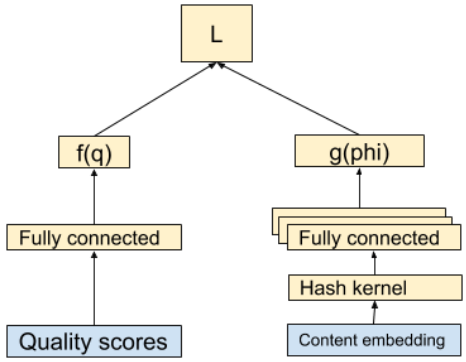

Figure 4: Architecture example for a deep DPP kernel.

To learn all of the parameters of the neural network for computing f and $\boldsymbol{g}$, we optimize LogLike from Equation 11 using Tensorflow [1]. The resulting deep DPP models have already shown utility improvements in live experiments (See the Deep DPPs entry in Table 1). However, these deeper models change the ranking substantially enough from the un-diversified baseline that secondary business metrics begin to be significantly impacted, requiring additional tuning.

4.5 Efficient Ranking Algorithm with DPP

In this section, we describe how we use at serving time the DPP parameters that were learned as described in Section 4.3 or Section 4.4. That is, when a user goes to the YouTube mobile homepage, how does the DPP decide which videos go at the top of their recommendations feed? For any given user, the underlying parts of the YouTube system infrastructure send to the DPP layer of the system the personalized quality scores q and video embedding vectors ϕ for a set of N videos. We construct a DPP kernel L from these scores and embeddings and the learned parameters as described in the previous section. We then fix some window size $k \ll N$, and we ask the DPP for a high-probability set of k videos. We put these videos at the top of the feed, then again ask the DPP for a high-probability set of k videos from the remaining $N - k$ unused videos. These videos become the next k in the feed. We repeat this process until we have ordered the entire feed of N videos.

The idea behind constructing sub-windows of the data with stride size k is that the repulsion between two similar items reduces as the distance between them in the feed increases. That is, having video 1 and video 100 be similar is not as detrimental to user enjoyment as having video 1 and video 2 be similar. In practice, for ordering a feed where N consists of several hundred videos, we use sub-windows where k is a dozen or so videos.

When we "ask the DPP for a high-probability set of k videos", what we are actually doing is asking for the size-k set Y that has the highest probability that the user interacts with *every* one of those k items.[1] This corresponds to the following maximization

[1] One could consider alternative quantities, like the probability that a user interacts at least one item in a given subset. We plan to consider such alternative formulations in future work.

Algorithm 1 Rank a video feed via a DPP

Require: α, σ ▷ DPP parameters
Require: $k \in N$ ▷ The size of the windows
Require: q, ϕ ▷ The quality estimates and video embeddings
Require: $W \leftarrow \{1, 2, \ldots, N\}$ ▷ Indices of the items to rank
  Construct D from ϕ ▷ Compute distances from embeddings
  $L \leftarrow N \times N$-matrix ▷ Construct the DPP
  for $i = 1, 2, \ldots, N$ **do**
    for $j = 1, 2, \ldots, N$ **do**
      $L_{ij} \leftarrow L_{W[i]W[j]}(\alpha, \sigma, q, D)$
    end for
  end for

  $R \leftarrow []$ ▷ The final ordering
  while $|W| > 0$ **do**
    $M \leftarrow \text{GreedyApproxMax}(L, \min(k, |W|))$ ▷ Indices into W
    $D \leftarrow \emptyset$
    for $i \in M$ **do**
      $R \leftarrow R + W[i]$ ▷ Get items from the greedy approx
      $D \leftarrow D + W[i]$
    end for
    $W \leftarrow W \setminus D$ ▷ Remove the selected items
    $L \leftarrow L[W, W]$ ▷ Restrict L to the W submatrix
  end while
  return R

problem:

$$\max_{Y:|Y|=k} \det(L_Y) \,. \tag{14}$$

As shown in [18], this maximization is NP-hard. In practice though, a standard greedy algorithm for submodular maximization from [31] seems to work well for approximately solving this problem. The greedy algorithm starts from $Y = \emptyset$ (the empty set), then runs k iterations, adding one video to Y on each iteration. The chosen video in iteration i is the video v that produces the largest determinant value when added to the current chosen set:

$$\max_{v \in \text{remaining videos}} \det(L_{Y \cup v}) \,. \tag{15}$$

Beyond its simplicity, an additional advantage of using this greedy algorithm is that, if we keep track of the order in which greedy selects videos, then this gives us a natural order for the videos in the corresponding size-k window of the user's feed.

Algorithm 1 summarizes the ranking algorithm described in this section. As we will see in the subsequent section, this ranking helps users find the content that they want to consume more easily.

5 EXPERIMENTAL RESULTS

First, we will describe some basic comparison baselines. Before finally arriving at DPPs, we tried three diversification heuristics:

(1) Fuzzy deduping: disallow any video i whose distance to a video j already in the feed is below a threshold τ: $D_{ij} < \tau$.
(2) Sliding window: allow at most n out of every m items to be below a distance threshold τ.
(3) Smooth score penalty: When selecting the video v for position $n+1$, re-scale the quality scores to account for similarity to videos 1 through n that have already been selected:

$$q_{\text{new},v} = q_{\text{original},v} * e^{-b(\phi_v \cdot \phi_{\text{previous}})} \tag{16}$$

$$\text{with } \phi_{\text{previous}} = \sum_{k=0}^{n} a^{n-k-1} \phi_k, \tag{17}$$

where q is the quality score we sort by, a and b are free parameters, and ϕ is the embedding vector.

Strategy	Satisfied homepage watchers
Fuzzy deduping	-0.05%
Sliding window	-0.26%
Smooth score penalty	-0.41%
DPPs	+0.63%
Deep DPPs	+1.72%

Table 1: Experimental results over approximately 1 week for various attempts at improving video diversity in users' feeds. "Satisfied homepage watchers" refers to a metric that considers sessions originating from the homepage and counts how many of these sessions were of significant duration—a notion of homepage utility. Only DPPs improved this metric.

As seen in Table 1, all of these attempts led to a less useful mobile homepage feed, as measured by the number of users with long sessions originating from homepage.

When experimenting with DPPs, we first used the kernel L described in Section 4.2, and evaluated a variety of embeddings and distance functions (dense and sparse audio embeddings, frame embeddings, thumbnail image embeddings, document text embeddings, etc.). We found that it works quite well to use Jaccard distances for D_{ij} in Equation 10, applied to sparse vectors ϕ consisting of item tokens. (For example, the Saturday Night Live video "Olive Garden -SNL" has tokens "snl", "olive garden", "saturday night", "night live", and "sketch", among others.) Live experiments on YouTube's mobile homepage recommendations saw dramatic improvements for our users. In addition to the +0.63% on the satisfied homepage watchers metric shown in Table 1, we also saw +0.52% in overall watch time, which is quite a significant jump over the baseline. Because of this success on mobile, diversification via DPPs has been deployed on all surfaces, including TV, desktop, and Live streams. (Note that while the deep Gramian DPPs system looks very promising on the "satisfied homepage watchers" metric, it has not yet been deployed. As mentioned earlier, these deeper models change the ranking substantially enough from the un-diversified baseline that secondary business metrics begin to be significantly impacted, requiring additional tuning.)

Interestingly, for some choices of parameters we saw losses in direct interactions *on* the homepage, though across the site we had an overall win. Figure 5 shows the percent increase in view time that *originates* from the homepage. This suggests that users find content sufficiently attractive that it leads to longer sessions starting from the homepage. And indeed, we did observe increased activity on the related videos panel (a panel of videos one sees alongside the video that is currently playing), in terms of click-through rate, number of views, and amount of total view time, despite the fact that our original change only affected the videos shown on the homepage

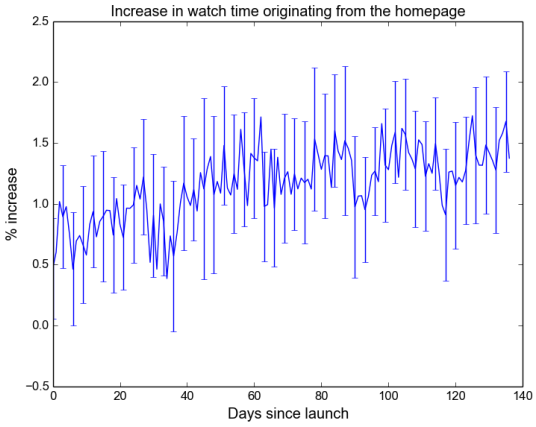

Figure 5: Although diversification did little to increase interactions directly on the homepage, it did increase the total *originating* from the homepage (taking the related video panel into account). Error bars represent 95% confidence intervals.

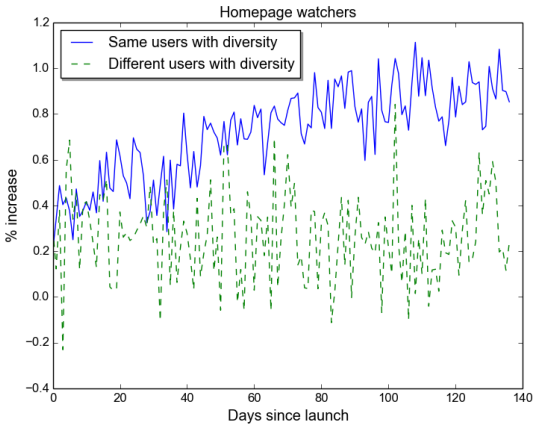

Figure 6: Evidence of a long-term "learning effect" as seen in the number of people watching videos from the homepage. The implication is a much more useful and satisfying product experience.

feed. Cumulatively, it suggests that users find more videos that they enjoy compared to before.

Moreover, we have been able to observe a long-term "learning effect" [17] from diversifying users' feeds. That is, diversification results in users returning to and enjoying our service more as time goes on. We measured this effect by running two sets of long-term holdback experiments[2]. In the first holdback condition, users do not get DPP-diversified feeds, but that subset of the user population changes every day (these users are normally exposed to the diversified feed, except on the rare day that they end up in this holdback set). In the second holdback condition, a consistent set of users do not see DPP-diversified feeds. We can then observe whether DPP diversification results in a long-term improvement in user experience by observing the difference between the two holdbacks when compared to their respective control groups. As we can see in Figure 6, which shows the increase in number of users watching at least one video from the homepage against these two holdback groups, users who have been exposed to diversified feeds more often realize that they can find videos of interest on YouTube's homepage. Therefore, we can say that diversified feeds lead to increased user satisfaction in the immediate term, and that this effect becomes even more pronounced over time.

[2]A holdback is simply an A/B experiment where users in group B do not receive the launched treatment.

6 CONCLUSIONS AND FUTURE WORK

Researchers realized well over a decade ago that diversification is an important problem for recommendation systems, and for information retrieval in general. Significant research efforts have invested in approaches that use a taxonomic or category-based approach, often combined with a variety of heuristics. In contrast, we propose using a method based on determinantal point processes (DPPs). Our approach performs set-wise optimization of recommendations. Since this approach naturally factors the problem into one of estimating item quality, and another of estimating repulsive effects between pairs of items, our stacked architecture allows us to leverage existing sophisticated investments in pointwise scoring and item analysis.

In this paper, we discussed the challenges of applying DPPs in a large-scale video recommendation system. We considered several parameterizations of the DPP kernel as well as learning methods for computing the value of the kernel parameters from positive user interactions with videos. Finally, we presented live experiment results on this large-scale system, showing both an immediate short-term lift in user utility, as well as long-term effects—users looked to YouTube more often to satisfy their needs.

Our work is not without limitations. First, the DPP we trained is non-personalized in that the parameters such as σ are learned from training on a large population of user data, not on a single user's data. In the near future, we hope to develop new approaches to understand each individual user's short-term and long-term diversification needs. We also do not fully understand how different domains or genres might affect diversification policy. For instance, users might prefer music videos to stay within a certain boundary (no vocals, for instance), as they might be enjoyed somewhat more passively, while genres like comedy might need more diversity. Additionally, we do not have a good model that takes time into account, such as understanding weekday vs. weekend diversity preferences. We would like to explore the connection of diversification methods with reinforcement learning, so that we can learn a good control policy for diversification. Given the multitude of directions for future work, we feel that our current work simply "scratches the surface" of the possibilities available to improve user experiences by moving away from pointwise estimators in recommender systems.

REFERENCES

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike

Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2015. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. (2015). http://tensorflow.org/ Software available from tensorflow.org.

[2] Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying Search Results. In *Conference on Web Search and Data Mining (WSDM)*. http://doi.acm.org/10.1145/1498759.1498766

[3] R. Bardenet and M. Titsias. 2015. Inference for Determinantal Point Processes Without Spectral Knowledge. In *Neural Information Processing Systems (NIPS)*.

[4] A. Borodin. 2009. Determinantal point processes. *ArXiv e-prints* (2009). https://arxiv.org/abs/0911.1153

[5] Jaime Carbonell and Jade Goldstein. 1998. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *Conference on Research and Development in Information Retreival (SIGIR)*. http://doi.acm.org/10.1145/290941.291025

[6] Olivier Chapelle, Shihao Ji, Ciya Liao, Emre Velipasaoglu, Larry Lai, and Su-Lin Wu. 2011. Intent-based Diversification of Web Search Results: Metrics and Algorithms. *Information Retrieval* 14, 6 (2011), 572–592. http://dx.doi.org/10.1007/s10791-011-9167-7

[7] Laming Chen, Guoxin Zhang, and Hanning Zhou. 2017. Improving the Diversity of Top-N Recommendation via Determinantal Point Process. In *Large Scale Recommendation Systems Workshop at the Conference on Recommender Systems (RecSys)*. http://arxiv.org/abs/1709.05135

[8] Charles L.A. Clarke, Maheedhar Kolla, Gordon V. Cormack, Olga Vechtomova, Azin Ashkan, Stefan Büttcher, and Ian MacKinnon. 2008. Novelty and Diversity in Information Retrieval Evaluation. In *Conference on Research and Development in Information Retreival (SIGIR)*. http://doi.acm.org/10.1145/1390334.1390446

[9] Paul Covington, Jay Adams, and Emre Sargin. 2016. Deep Neural Networks for YouTube Recommendations. In *Conference on Recommender Systems (RecSys)*.

[10] Van Dang and W. Bruce Croft. 2012. Diversity by Proportionality: An Election-based Approach to Search Result Diversification. In *Conference on Research and Development in Information Retreival (SIGIR)*. http://doi.acm.org/10.1145/2348283.2348296

[11] Marina Drosou and Evaggelia Pitoura. 2010. Search Result Diversification. *SIGMOD Record* 39, 1 (2010), 41–47. http://doi.acm.org/10.1145/1860702.1860709

[12] Mike Gartrell, Ulrich Paquet, and Noam Koenigstein. 2016. Bayesian Low-Rank Determinantal Point Processes. In *Conference on Recommender Systems (RecSys)*.

[13] J. Gillenwater. 2014. *Approximate Inference for Determinantal Point Processes*. Ph.D. Dissertation. University of Pennsylvania.

[14] J. Gillenwater, A. Kulesza, E. Fox, and B. Taskar. 2014. Expectation-Maximization for Learning Determinantal Point Processes. In *Neural Information Processing Systems (NIPS)*.

[15] Sreenivas Gollapudi and Aneesh Sharma. 2009. An Axiomatic Approach for Result Diversification. In *Conference on the World Wide Web (WWW)*. http://doi.acm.org/10.1145/1526709.1526761

[16] Yoshinori Hijikata, Takuya Shimizu, and Shogo Nishida. 2009. Discovery-oriented Collaborative Filtering for Improving User Satisfaction. In *Conference on Intelligent User Interfaces (IUI)*. http://doi.acm.org/10.1145/1502650.1502663

[17] Henning Hohnhold, Deirdre O'Brien, and Diane Tang. 2015. Focus on the Long-Term: It's better for Users and Business. In *Conference on Knowledge Discovery and Data Mining (KDD)*. http://dl.acm.org/citation.cfm?doid=2783258.2788583

[18] Chun-Wa Ko, Jon Lee, and Maurice Queyranne. 1995. An Exact Algorithm for Maximum Entropy Sampling. *Operations Research* 43, 4 (1995), 684–691. http://www.jstor.org/stable/171694

[19] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37. http://dx.doi.org/10.1109/MC.2009.263

[20] Alex Kulesza and Ben Taskar. 2011. k-DPPs: Fixed-Size Determinantal Point Processes. In *International Conference on Machine Learning (ICML)*.

[21] Alex Kulesza and Ben Taskar. 2011. Learning Determinantal Point Processes. In *Conference on Uncertainty in Artificial Intelligence (UAI)*.

[22] Alex Kulesza and Ben Taskar. 2012. Determinantal Point Processes for Machine Learning. *Foundations and Trends in Machine Learning* 5, 2-3 (2012), 123–286. http://dx.doi.org/10.1561/2200000044

[23] YoungOk Kwon and Gediminas Adomavicius. 2007. New Recommendation Techniques for Multicriteria Rating Systems. *IEEE Intelligent Systems* 22 (2007), 48–55.

[24] Neal Lathia, Stephen Hailes, Licia Capra, and Xavier Amatriain. 2010. Temporal Diversity in Recommender Systems. In *Conference on Research and Development in Information Retreival (SIGIR)*. http://doi.acm.org/10.1145/1835449.1835486

[25] Hui Lin and Jeff Bilmes. 2011. A Class of Submodular Functions for Document Summarization. In *Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT)*. http://dl.acm.org/citation.cfm?id=2002472.2002537

[26] Zelda Mariet and Suvrit Sra. 2015. Fixed-Point Algorithms for Learning Determinatal Point Processes. In *International Conference on Machine Learning (ICML)*.

[27] Zelda Mariet and Suvrit Sra. 2016. Kronecker Determinantal Point Processes. In *Neural Information Processing Systems (NIPS)*.

[28] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring Networks of Substitutable and Complementary Products. In *Conference on Knowledge Discovery and Data Mining (KDD)*. http://doi.acm.org/10.1145/2783258.2783381

[29] Sean M. McNee, John Riedl, and Joseph A. Konstan. 2006. Being Accurate is Not Enough: How Accuracy Metrics Have Hurt Recommender Systems. In *CHI Extended Abstracts on Human Factors in Computing Systems*. http://doi.acm.org/10.1145/1125451.1125659

[30] H. Nassif, K.O. Cansizlar, M. Goodman, and S.V.N. Vishwanathan. 2016. Diversifying Music Recommendations. In *International Conference on Machine Learning (ICML) Workshop*.

[31] G. Nemhauser, L. Wolsey, and M. Fisher. 1978. An Analysis of Approximations for Maximizing Submodular Set Functions I. *Mathematical Programming* 14 (1978), 265–294.

[32] Yonathan Perez, Michael Schueppert, Matthew Lawlor, and Shaunak Kishore. 2015. Category-Driven Approach for Local Related Business Recommendations. In *Conference on Information and Knowledge Management (CIKM)*. 73–82. http://dl.acm.org/citation.cfm?doid=2806416.2806495

[33] Davood Rafiei, Krishna Bharat, and Anand Shukla. 2010. Diversifying Web Search Results. In *Conference on the World Wide Web (WWW)*. http://doi.acm.org/10.1145/1772690.1772770

[34] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. 1994. GroupLens: An Open Architecture for Collaborative Filtering of Netnews. In *Conference on Computer Supported Cooperative Work (CSCW)*. http://doi.acm.org/10.1145/192844.192905

[35] Rodrygo L.T. Santos, Craig Macdonald, and Iadh Ounis. 2010. Exploiting Query Reformulations for Web Search Result Diversification. In *Conference on the World Wide Web (WWW)*. http://doi.acm.org/10.1145/1772690.1772780

[36] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based Collaborative Filtering Recommendation Algorithms. In *Conference on the World Wide Web (WWW)*. http://doi.acm.org/10.1145/371920.372071

[37] Choon Hui Teo, Houssam Nassif, Daniel Hill, Sriram Srinivasan, Mitchell Goodman, Vijai Mohan, and S.V.N. Vishwanathan. 2016. Adaptive, Personalized Diversity for Visual Discovery. In *Conference on Recommender Systems (RecSys)*. http://doi.acm.org/10.1145/2959100.2959171

[38] Sebastian Tschiatschek, Adish Singla, and Andreas Krause. 2017. Selecting Sequences of Items via Submodular Maximization. In *Conference on Artificial Intelligence (AAAI)*.

[39] Saúl Vargas, Linas Baltrunas, Alexandros Karatzoglou, and Pablo Castells. 2014. Coverage, Redundancy and Size-awareness in Genre Diversity for Recommender Systems. In *Conference on Recommender Systems (RecSys)*. http://doi.acm.org/10.1145/2645710.2645743

[40] Erik Vee, Utkarsh Srivastava, Jayavel Shanmugasundaram, Prashant Bhat, and Sihem Amer Yahia. 2008. Efficient Computation of Diverse Query Results. In *International Conference on Data Engineering (ICDE)*. http://dx.doi.org/10.1109/ICDE.2008.4497431

[41] Cong Yu, Laks Lakshmanan, and Sihem Amer-Yahia. 2009. It Takes Variety to Make a World: Diversification in Recommender Systems. In *Conference on Extending Database Technology (EDBT)*. http://doi.acm.org/10.1145/1516360.1516404

[42] Cheng Xiang Zhai, William W. Cohen, and John Lafferty. 2003. Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. In *Conference on Research and Development in Information Retreival (SIGIR)*. http://doi.acm.org/10.1145/860435.860440

[43] Mi Zhang and Neil Hurley. 2008. Avoiding Monotony: Improving the Diversity of Recommendation Lists. In *Conference on Recommender Systems (RecSys)*. http://doi.acm.org/10.1145/1454008.1454030

[44] Jiaqian Zheng, Xiaoyuan Wu, Junyu Niu, and Alvaro Bolivar. 2009. Substitutes or Complements: Another Step Forward in Recommendations. In *Conference on Electronic Commerce (EC)*. http://doi.acm.org/10.1145/1566374.1566394

[45] Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. 2005. Improving Recommendation Lists Through Topic Diversification. In *Conference on the World Wide Web (WWW)*. http://doi.acm.org/10.1145/1060745.1060754