# Fingerprinting Ratings For Collaborative Filtering — Theoretical and Empirical Analysis

Yoram Bachrach, Ralf Herbrich

Microsoft Research

**Abstract.** We consider fingerprinting methods for collaborative filtering (CF) systems. In general, CF systems show their real strength when supplied with enormous data sets. Earlier work already suggests sketching techniques to handle massive amounts of information, but most prior analysis has so far been limited to non-ranking application scenarios and has focused mainly on a theoretical analysis. We demonstrate how to use fingerprinting methods to compute a *family* of rank correlation coefficients. Our methods allow identifying users who have similar rankings over a certain set of items, a problem that lies at the heart of CF applications. We show that our method allows approximating rank correlations with high accuracy and confidence. We examine the suggested methods empirically through a recommender system for the Netflix dataset, showing that the required fingerprint sizes are even smaller than the theoretical analysis suggests. We also explore the of use standard hash functions rather than min-wise independent hashes and the relation between the quality of the final recommendations and the fingerprint size.

## 1 Introduction

Recommender systems supply users with items they are likely to find interesting. Some methods use the content of the information item (in the *content based approach*). We focus on the alternative *collaborative filtering approach* (CF systems), where the system predicts whether an item is likely to interest the target user, based on the ranking of that item by other users. One obstacle in constructing real-world CF systems is the need to handle huge volumes of information.

Previous work [7] suggested a technique for computing the similarity between users, based on *sketching* — rather than storing the full lists of items for each user, it stores a concise *fingerprint* of the lists of examined items, called a *sketch*. These fingerprints are extremely short, much shorter than compression techniques allow, but only allow specific computations on the data. The sketches of [7] allow approximating the *proportional intersection similarity* (PI) of any two users. This method has been extended in [6], where similar fingerprints were used to compute the correlation between two user's rankings of items.

Both [7, 6] have significant shortcomings. First, they focused on a *very specific* rank correlation coefficient — Kendall's Tau [19]. Other correlations, such as Spearman's rank correlation [23], are more appropriate for some settings [15]. For

example, Spearman's Rho has the meaningful interpretation as a Pearson correlation coefficient, and known statistical tests can use it in significance testing. Second, their sketches use *min-wise independent families of hashes* (MWIFs). MWIFs are hard to construct and slow to use. Third, they only analyze sketches *theoretically*, lacking empirical evidence regarding the quality of the sketches in terms of the quality of the *final recommendations* based on these approximations. Our contributions are:

1. We suggest a similar fingerprint which allows computing a *family* of rank correlation coefficients, including the prominent *Spearman rank correlation*.
2. We discuss *empirical analysis* of such techniques, based on *Collabriprint*, our CF infrastructure which uses fingerprinting techniques, that wastested on the Netflix [8] dataset. Our empirical analysis shows that in practice:
   - It suffices to use smaller sketches than the theoretical results require.
   - It is possible to use *standard* hash functions, such as MD5 [21], instead of the more complex MWIFs, and still obtain high accuracy and confidence.
   - The final recommendation's quality depends on the fingerprints' size. Even small fingerprints result in high quality recommendations.

## 2 Preliminaries

We first briefly explain the problem of fingerprinting in CF systems. Consider Alice and Bob, who have *both* examined a set of $n$ items. In some CF domains, the mere fact that a user has examined an item implicitly tells the CF system that the user liked the item. In other domains, explicit information is available as users rate examined items on a certain scale. CF systems first seek users who share similar rating patterns with the target user, and use their ratings to generate a prediction for how the target user would rate items she has not examined. A method for approximating the *Proportional Intersection (PI)* was suggested in [7]. Given two users, Alice and Bob who examined the *same* number of items, their PI is defined as follows. Denote by $C_i$ the set of items Alice examined, and by $C_j$ the set of items Bob examined. Both users examined the same number of items, so $|C_i| = |C_j|$. The PI is $\frac{|C_i \cap C_j|}{|C_i|} = \frac{|C_i \cap C_j|}{|C_j|}$. The Jackard measure is a similar measure when $|C_i| \neq |C_j|$, and is defined as $J_{i,j} = \frac{|C_i \cap C_j|}{|C_i \cup C_j|}$.

The PI and Jackard measures only consider *which* items were examined. *Rank correlations*, such as Spearman's Rho and Kendall's Tau, measure the similarity between two rankings (orderings) of the same items. Spearman's Rho is simply a special case of the Pearson product-moment coefficient, in which the data sets are converted to rankings before calculating the coefficient. Let $x_i = r_a(i)$ and $y_i = r_b(i)$ be the rankings of item $i$, given by Alice and Bob, and let $d_i = x_i - y_i$. Spearman's Rho $\rho_{r_a,r_b}$ can be computed using the following direct formula: $\rho_{r_a,r_b} = 1 - \frac{6\sum_{i=0}^{n} d_i^2}{n(n^2-1)}$. Both Kendall's Tau and Spearman's Rho range from -1 (strong negative correlation) to 1 (strong positive correlation).

Computing user similarity metrics allow constructing CF recommender systems, by predicting the rating a target user would give any unexamined item,

based on the ratings given by other users, weighted according to similarity to the target user. User similarity can be computed using the full information, consisting of the lists of examined items and their ratings for each user. However, such data sets can be extremely large, so it is desirable to compute similarities while minimizing the size of the data. Fingerprinting provides a good tradeoff between the required storage and the quality of the predictions.

The method of [7] approximates the PI $p_{i,}$ for any two users $i, j$, by maintaining short fingerprints of the lists of examined items, called *sketches*. This method assumes $i$ and $j$ have equal size lists of items $C_1, C_2$ (so $|C_i| = |C_j| = n$), and the size of each sketch depends on the target confidence $\delta$ and accuracy $\epsilon$. The method returns an approximation $\hat{p}_{i,j}$ to $p_{i,j}$ such that, with probability of at least $1 - \delta$, $|p_{i,j} - \hat{p}_{i,j}| \leq \epsilon$. Building on this work, a method for computing the *Kendall Tau correlation* was proposed in [6]. This improves recommendations, since even users who examined similar items may rate them differently.

Similarly to the above techniques, we also use a Min-Wise Independent Family of hashes (MWIF). Let $H$ be a family of functions over the source $X$ and target $Y$, so each $h \in H$ is a function $h : X \to Y$, where $Y$ is completely ordered. We say that $H$ is MWIF if when randomly choosing a function $h \in H$, for any subset $C \subseteq X$, any $x \in C$ has an equal probability to be minimal under $h$.[1]. Formally, we say that $H$ is MWIF, if for all $C \subseteq X$, for any $x \in C$, $Pr_{h \in H}[h(x) = min_{a \in C} h(a)] = \frac{1}{|C|}$. MWIF computations are slow, making them ill-suited for many practical applications. For full discussion of MWIFs and their construction see [10, 17].

## 3 Rank Correlation Fingerprints

Let $i, j$ be two users, and $C_i, C_j$ the set of items each has examined. We now present our fingerprinting method, based on randomly choosing hashes $h$ from a MWIF $H$. Similarly to [7], we consider the identities of items in the set $C_i$ of items examined by each user as integers, apply $h$ to all these integers and examine the minimal value obtained. Given a randomly chosen $h \in H$ we denote minimal value obtained after applying $h$ to all elements in $C_i$ as $m_i^h = min_{x \in C_i} h(x)$. Performing the same on $C_j$ we denote $m_j^h = min_{x \in C_j} h(x)$. We now examine the probability that $m_i^h = m_j^h$. Theorem 1 in [7] has shown that when $|C_i| = |C_j|$ so the PI is $p_{i,j} = \frac{C_i \cap C_j}{|C_i|} = \frac{C_i \cap C_j}{|C_j|}$, we have $Pr_{h \in H}[m_i^h = m_j^h] = \frac{p_{i,j}}{2 - p_{i,j}}$. We provide a similar proof for the Jackard measure, $J_{i,j} = \frac{|C_i \cap C_j|}{|C_i \cup C_j|}$.

**Theorem 1.** $Pr_{h \in H}[m_i^h = m_j^h] = J_{i,j}$.

*Proof.* Denote $x = J_{1,2}$. The set $C_i \cup C_j$ contains three types of items: items that appear *only* in $C_i$, items that appear *only* in $C_j$, and items that appear in $C_i \cap C_j$. When an item in $C_i \cap C_j$ is minimal under $h$, i.e., for some $a \in C_i \cap C_j$

[1] It does not matter which distribution is used to choose $h$ from $H$, as long as this distribution makes $H$ a MWIF.

we have $h(a) = min_{x \in C_1 \cup C_2} h(x)$, we get that $min_{x \in C_i} h(x) = min_{x \in C_j} h(x)$. On the other hand, if for some $a \in C_i \cup C_j$ such that $a \notin C_i \cap C_j$ we have $h(a) = min_{x \in C_1 \cup C_2} h(x)$, the probability that $min_{x \in C_i} h(x) = min_{x \in C_j} h(x)$ is negligible [2]. Since $H$ is MWIF, any element in $C = C_i \cup C_j$ is equally likely to be minimal under $h$. However, only elements in $I = C_i \cap C_j$ would result in $m_i^h = m_j^h$. Thus $Pr_{h \in H}[m_i^h = m_j^h] = \frac{1}{|C_i \cup C_j|} \cdot |C_i \cap C_j| = \frac{|C_i \cap C_j|}{|C_i \cup C_j|} = J_{i,j}$.

The fingerprints used in [7] are called *item sketches*, and are created using $k$ hash functions. Let $v_k = \langle h_1, h_2, \ldots, h_k \rangle$ be a tuple of $k$ randomly chosen functions from the MWIF $H$, and let $C_i$ be the set of items that user $i$ has examined. Denote the minimal item in $C_i$ under $h_s$ as $m_i^{h_s} = min_{x \in C_i} h_s(x)$.

**Definition 1 (Item Sketches).** *The $H_k$ sketch of $C_i$, $S(C_i)$, is the list of minimal items in $C_i$ under the $k$ randomly chosen functions from $h$: $S^k(C_i) = (m_i^{h_1}, m_i^{h_2}, \ldots, m_i^{h_k})$.*

We call a hash $h_s$ where $m_i^{h_s} = m_j^{h_s}$ a collision hash, and say location $s$ is a *sketch collision* for $i, j$. The work [7] shows that in order to approximate the PI $p_{a,b}$ with accuracy $\epsilon$ and confidence $\delta$, it is enough to use $k = \frac{\ln \frac{2}{\delta}}{2 \frac{\epsilon^2}{9}}$ hashes. However, they do not compute how well the users' tastes correlate, a problem later addressed in [6] where the Kendall Tau correlation is approximated. We focus on a different family of correlations, based on Spearman's Rho. CF systems seek users similar to a target user, filtering out users with a low Jackard similarity to that user. Similarly to [6] we assume the CF system filters out any user with a Jackard score (or PI score) lower than some value $p^*$, and augment the item sketches to compute rank correlations. The system then recommends items based on scores that weight rankings given by users according to their similarity with the target user. A strong user similarity metric is rank correlation.

Our fingerprints are the *item sketches* of Definition 1, augmented with the *rating* of the minimal item under the hash. Consider Alice and Bob, with Jackard similarity of at least $p^*$. The item sketches in Definition 1 use $k$ random hashes, and the fingerprint is the list of the minimal items under each hash. Due to Theorem 1, given users $i$ with items $C_i$ and $j$ with items $C_j$, the probability of a collision for $i, j$ on any location $s$ (i.e. $P(m_i^{h_s} = m_j^{h_s})$) depends on $J_{i,j}$. Due to Theorem 1, if $J_{i,j} \geq p^*$, any location has a probability of at least $p^*$ of being a collision. A collision in location $s$ is $h_s(q)$, where $q$ is an identity of an item chosen uniformly at random from $C_i \cap C_j$ (an item both $i$ and $j$ examined). Our fingerprints include the rating of the item $q$ [3].

Similarly to item sketches (Definition 1), each location is built using a randomly chosen hash. Let $h_i$ be the hash for the $i$'th location. The augmentation

[2] Such an event requires that two *different* items, $x_i \in C_i$ and $x_j \in C_j$ to be mapped to the same value $h^* = h(x_i) = h(x_j)$, and that this value would also be the minimal value obtained when applying $h$ to both all items in $C_i$ and in $C_j$. As discussed in [17], the probability for this is negligible when $h$'s range is large enough.

[3] The sketches of [6] are similar, although we employ a very different algorithm to compute Spearman's Rho (whereas they compute Kendall's Tau).

for location $i$ contains the *rating* of the item that is minimal under $h_i$. When constructing the sketch for user $a$, we consider the user's item set $C_a$ and the ratings of the items in $C_a$. The rating of user $a$ for items in $C_a$ is denoted as $r_a$. Thus, $r_a$ maps items in $C_a$ to their rating. Given $h_i$, consider the set of items that are minimal under $h_i$ [4], i.e. $M = \{x \in C_a | h_i(x) = m_a^{h_i}\}$. If only one item is minimal under the hash, so $|M| = 1$, we denote $M = \{m\}$, and denote the rating of that item as $g_a^i = r_a(m)$. Only with a very low probability do we have $|M| > 1$. If $|M| > 1$, denote $m'$ to be the minimal item in $M$, under some predetermined ordering (not under $h_i$), and denote $g_a^i = r_a(m')$. The sketch for user $a$ in the $i$'th location contains the minimal item in $C_a$ under $h_i$, and its rating in $a$'s eyes. We denote the sketch for user $a$ with items $C_a$ (where the sketch is based on $H_k = \langle h_1, \ldots, h_k \rangle$, the $k$ randomly chosen hashes from the MWIF), as $S^k(C_a)$.

**Definition 2 (Rank Correlation (RC) Sketches).** *The $H_k$ RC sketch of $C_a$, $S^k(C_a)$, contains the both the item sketch and the rank sketch. The item sketch is the list of minimal items in $C_a$ under the $k$ randomly chosen hash functions from, so $S^k_{items}(C_a) = (m_a^{h_1}, m_a^{h_2}, \ldots, m_a^{h_k})$, and the rank sketch contains the ranks of these items, so $S^k_{ranks}(C_a) = (g_a^1, \ldots g_a^n)$. The rank correlation sketch is the concatenation of these two sketches.*

The fingerprint size required for approximating Kendall's Tau using RC sketches was analyzed in [6]. We provide a similar analysis for a Spearman's Rho. Observe that an RC sketch collision for two users [5] provides the ratings of each of the two users of a *randomly chosen* item from $C_a \cap C_b$ . Thus, a collision provides $r_a(x), r_b(x)$ for a randomly chosen items $x \in C_a \cap C_b$. We now determine *how many* collisions are required to approximate Spearman's Rho with a target accuracy and confidence. We wish to return an approximation $\hat{\rho_{a,b}}$ to Spearman's Rho $\rho_{a,b}$ such that with probability of at least $1 - \delta$ we have $|\rho_{r_i,r_j} - \hat{\rho_{i,j}}| \leq \epsilon$. We use the following theorem from [6] regarding the required number of hashes to provide at least $k$ collisions.

**Theorem 2.** *Let $k$ be a certain required sketch collisions, and let $p$ be a bound from below on the Jackard similarity of any two users. The required fingerprint size to achieve the required number $k$ of sketch collisions with probability $1 - \delta_c$ is $m \geq \frac{k}{p} + \frac{\ln \frac{1}{\delta_c}}{4p^2}(1 + 3\sqrt{k})$.*

The sketch collision probability depends on the Jackard similarity (as shown in Theorem 1). Theorem 2 shows that given a minimal Jackard similarity, a long enough fingerprint would provide the required number of collisions with

[4] If each item is hashed to a different value then there is only one item whose value under the hash is minimal. However, several items may be mapped to the same value, so there may be several items minimal under the hash.

[5] Recall that a sketch collision is a sketch location $i$ with hash function $h_i$ where the minimal items of the two users ($a$ with items $C_a$ and $b$ with items $C_b$) under $h_i$ are the same, so $m_a^{h_i} = m_b^{h_i}$.

high probability. The required fingerprint length is logarithmic in the required confidence $\delta_c$ and polynomial in the required number of collisions. We now show that a family of rank correlations, including Spearman's Rho, can be computed using the RC sketches of Definition 2, generalizing the results of [6].

We now show how the RC sketches from Definition 2 allow computing a family of rank correlations. Members of this family could be expressed as a certain bounded function of the rank differences, summed across all items. We begin by an analysis of Spearman's Rho, and then generalize to this family of rank correlations. The definition of Spearman's Rho had a direct formula for it: $\rho_{r_a,r_b} = 1 - \frac{6\sum_{i=0}^{n} d_i^2}{n(n^2-1)}$. We first note that $d_i$ is simply the difference between the rating of a certain item in the first user's eyes and in the second user's eyes. Consider an item $x$ chosen uniformly at random from the set of possible items. We can examine $r_a(x)$ and $r_b(x)$ and define the following random variable.

**Definition 3.** *The Spearman's Rho random variable $X_i$, for item $x$ is $X_i = 1 - \frac{6(r_a(x)-r_b(x))^2}{n^2-1}$*

The random variable $X_i$ has an expectation of: $\mathrm{E}[X_i] = \mathrm{E}[1-\frac{6(r_a(x)-r_b(x))^2}{n^2-1}] = 1 - \frac{6}{n^2-1}\mathrm{E}[(r_a(x)-r_b(x))^2] = 1 - \frac{6}{n^2-1} \cdot \frac{1}{n}(\sum_i (r_a(i)-r_b(i))^2 = 1 - \frac{6\sum_{i=0}^{n} d_i^2}{n(n^2-1)} = \rho_{r_a,r_b}$.

We now denote $\rho = \rho_{r_a,r_b}$ for short. Given $k$ such random variables, $X_1, \ldots, X_k$, we can use $\frac{1}{k}\sum_{i=1}^{k} X_i$ as an estimate for $\rho$. We now derive the required number of such random items to approximate $\rho$ with accuracy $\epsilon$ and confidence $\delta$. To achieve the desired accuracy and confidence, the number of sampled items, $k$, must be large enough. We find the appropriate $k$ by using Hoeffding's inequality [16] (see similar analysis for very different uses in [11, 4, 3]).

**Theorem 3 (Hoeffding's inequality).** *Let $X_1, \ldots, X_n$ be independent random variables, where all $X_i$ are bounded so that $X_i \in [a_i, b_i]$, and let $X = \sum_{i=1}^{n} X_i$. Then the following inequality holds.*

$$\Pr(|X - \mathrm{E}[X]| \geq n\epsilon) \leq 2\exp\left(-\frac{2\,n^2\,\epsilon^2}{\sum_{i=1}^{n}(b_i-a_i)^2}\right)$$

Let $X_1, \ldots, X_k$ be the series $k$ of random variables, as defined above. Let $X = \sum_{j=1}^{k} X_j$, and take $\hat{\rho} = \frac{X}{k}$ as an estimator for $\rho$.

**Theorem 4.** *A confidence interval for $\rho$ is $[\hat{\rho}-\epsilon, \hat{\rho}+\epsilon]$. This interval holds the correct $\rho$ with probability of at least $1-\delta$. The required number of pair samples to perform this is $k \geq \frac{18\ln\frac{2}{\delta}}{\epsilon^2}$*

*Proof.* We use Hoeffding's inequality to bound the error below the target confidence level $\delta$. We note that $0 \leq \frac{d_i^2}{n^2-1} \leq 1$. Due to the definition of $X_i$ (see Definition 3), all $X_i$ are bounded between -5 and 1, and $\mathrm{E}[X] = k \cdot \rho$. Thus, from Hoeffding's inequality, the following holds: $\Pr(|X - k\rho| \geq k\epsilon) \leq 2e^{-\frac{1}{18}\,k\,\epsilon^2}$. Therefore the following also holds: $\Pr(|\hat{\rho}-\rho| \geq \epsilon) \leq 2e^{-\frac{1}{18}\,k\,\epsilon^2}$. We get that $-\frac{1}{18}\,k\,\epsilon^2 \leq \ln\frac{2}{\delta}$. Finally we obtain: $\epsilon \geq \sqrt{\frac{18\ln\frac{2}{\delta}}{k}}$ and $k \geq \frac{18\ln\frac{2}{\delta}}{\epsilon^2}$.

Using fingerprints with a the length determined by Theorem 2, we have a high probability of getting a large enough number of sketch collisions. Each such sketch collision gives the rating $r_a(x), r_b(x)$ of a certain randomly chosen item $x$, that both users ($a$ and $b$) ranked. Thus, with high probability, we obtain a series of random variables as required by Theorem 4. To compute an estimate for Spearman's Rho, we take the rankings $r_a(x), r_b(x)$ of each item $x$ that occurs on a sketch collision, and use them to compute $X_i = 1 - \frac{6(r_a(x)-r_b(x))^2}{n^2-1}$, the random variables defined above. Given $c$ sketch collisions, as above, we use $\frac{1}{c}\sum_{i=1}^{c} X_i$ as an estimate for $\rho$. The analysis so far was specific for Spearman's Rho. However, we now show the same type of an analysis can be used for many similar rank correlation functions.

**Theorem 5.** *Let $a$ be a constant and the function $f$ be bounded between certain constant values $b_l$ and $b_h$. The previous fingerprinting approach can be used to compute any rank correlation of the form: $\alpha = a + \frac{1}{n}\sum_i f(r_a(i), r_b(i))$.*

*Proof.* Let $a$ be a constant and $f$ a function bounded between $b_l$ and $b_h$, and consider a rank correlation of the form defined above. We can define a set of random variables $X_i$ as in Definition 3. The expectancy of the $X_i$'s would be $\alpha$. Also, since $f$ is bounded, we can apply Hoeffding in the same way. Note the bound distance $|b_h - b_l|$ only changes the resulting constant in the expression derived for the fingerprint size, so the approach works well for any bounds. Performing the analysis similarly to Theorem 4 gives the fingerprint size for any member of this family of functions, and the same RC sketches can be used.

## 4 Empirical Analysis

We tested the CF fingerprinting approach by analyzing approximations of similarity metrics in the Netflix [8] movie ratings dataset. As discussed in the introduction, there are several disadvantages to the approaches of [7, 6]: the use of MWIFs, the high theoretical bound on the fingerprint length, and the lack of empirical evaluation regarding the quality of the similarity approximation and *final recommendations*. We discuss how to overcome these drawbacks, and support this with empirical evidence. We show how to replace MWIFs with MD5 [21], widely used hash function. We show that the accuracy of the procedure in practice is much higher than the theoretical bounds, and empirically investigate the relation between overall recommendation accuracy and the fingerprint length.

The Netflix dataset is a movie ratings dataset, released in October 2006 by Netflix (www.netflix.com) [8]. It contains a 100 million anonymous movie ratings, given by half a million users on a collection of 17,000 movies. Fingerprinting allows approximating user similarity with high accuracy. Our framework, called *Collabriprint* was built using C# and F#. We used it on the Netflix dataset, running several tests. We computed both movie to movie similarity through the PI/Jackard similarity of the sets of users who watched the movies, and rank correlation similarity through Kendall's Tau and Spearman's Rho correlation

between users's rating of movies. We have examined the approximation error in similarity and the change in recommendation quality for different fingerprint lengths, as measured by the number $k$ of hashes used. Our implementation has used various MD5 hash functions rather than MWIFs.

Although MWIF hashes are required for the theoretical results, constructing and using such a family is computationally expensive, and there are no widely used implementations of them. As an alternative, we have chosen to use the MD5 hash [21], a widely used hash. Since we require many such functions, we used HMAC (keyed Hash Message Authentication Code) versions of MD5, HMAC-MD5. HMACs are computed using a hash function in combination with a key, where different keys result in different hash functions, all of which appear to have a random behavior. We chose MD5 for several reasons: it is a cryptographic hash functions with semi-random behavior; It has an HMAC version; It is commonly used in many applications, and there are widely available libraries implementing it; It works quite quickly in terms of computation time.

Our first tests were conducted on randomly chosen movies pairs. For each pair we computed the Jackard similarity using the full data set, and through fingerprints. Denoting the correct PI as $p$ and the PI estimate as $\hat{p}$, the inaccuracy for the movie pair is $e = |p - \hat{p}|$. Given an accuracy level $\epsilon$ we say the experiment had a big error if $e \geq \epsilon$, and say it was accurate if $e < \epsilon$. Let $s$ be a sequence of $m$ experiments. Given $\epsilon$, denote by $b_\epsilon$ the number of experiments with a big error, and $g_\epsilon = m - b_\epsilon$ the number of accurate ones. We denote the fraction of bad experiments as $f_b(\epsilon) = \frac{b_\epsilon}{m}$. Let $\delta$ be a confidence level. The *empirical accuracy* for a target confidence $\delta$, is the maximal $\epsilon$ for which $f_b(\epsilon)$, the fraction of bad experiments, is at most $\delta$. For our analysis we used a confidence level $1-\delta = 0.9$. For each fingerprint size $s$, we chose 2000 random movie pairs. For each such pair we performed 10 experiments, each using a different fingerprint of $s$ random hash functions. Thus, for each fingerprint size we had $20,000$ experiments. We measured the empirical accuracy for that sequence. The theoretical fingerprint size for target accuracy $\epsilon = 0.1$ and target confidence $1-\delta = 0.9$ (from the bounds in [7]), is $s = 1350$. The required size for $\epsilon = 0.15$ and $1 - \delta = 0.9$ is $s = 600$. We tested the empirical accuracy $\epsilon_e(s)$ for fingerprint sizes of $15, 20, 25, \ldots, 100$ and of $150, 200, \ldots, 650$ (all of which are much shorter than the required size for accuracy $\epsilon = 0.1$ and $1 - \delta = 0.9$). Figure 1 shows the empirical accuracy (measured in the experiment sequence) and the theoretical accuracy (obtained from the theoretical formulas), for a confidence level of $1 - \delta = 0.9$. Lower accuracy numbers are better, as the accuracy is the maximal allowed error. Figure 1 shows that on the Netflix dataset, the actual accuracy is much better than the theoretical bounds predict.

We also attempted to find the required fingerprint size to achieve a certain target accuracy $\epsilon$ (with a target confidence of $1 - \delta = 0.9$). To get the empirical required fingerprint size $s_e$ for target accuracy $\epsilon$, we found the minimal fingerprint size $s$ such that the empirical accuracy $\epsilon_e$ for that size is better than the required accuracy $\epsilon$ (i.e. $\epsilon_e(s) < \epsilon$). The following figure presents both the theoretical and empirical required sizes for different values of target accuracy.

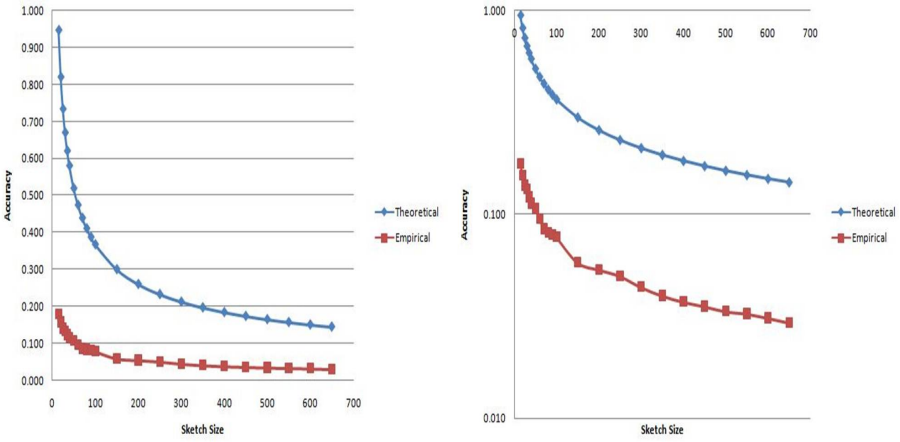


**Fig. 1.** Theoretical accuracy and empirical accuracy, for confidence 0.9 (right - logarithmic scale )

As Figure 2 shows, the required fingerprint size in empirical tests is much smaller than the theoretical bounds. The figure shows the empirical fingerprint size is roughly proportional to the theoretical bounds. The empirical size is about only 5% of the theoretical required size. The above results indicate that in practice it is not necessary to use large sizes to achieve very good accuracy. Given a dataset sample, we suggest finding the right size to use empirically.

We analyzed the quality of the recommendations based on fingerprints of different length. We implemented a simple recommendation algorithm, based on [9], where the score for item $i$ for target user $u$ (using the user set $U$ of recommender) is $\hat{u}+k\cdot\sum_{s\in U} sim(u,s)\cdot(s[i]-\hat{s})$ where $sim(u,s)$ is the similarity between $u, s$, such as Jackard, Spearman Rho or Kendall's Tau, $s[i]$ is the ranking user $s$ gives item $i$, and $\hat{u}$ is the average rating of user $u$. The value $k$ is used as a normalizing factor, typically $\frac{1}{\sum_{s\in U} sim(u,s)}$. Our recommender set $U$ was the 1000 most Jackard similar users, and we used Kendall Tau for $sim$. Both measures can be computed using the full data, or by fingerprinting. Obviously, the fingerprint scores differ from the full data scores.

Consider the scores computed for each movie in the full data set, which we call *true scores*. When ordering movies according to the true scores, the first items are the best recommendations. We call an item in the top 5% of the list *relevant items*. Now consider scores computed using the fingerprints only, which we call *fingerprint scores*. Sorting the list by fingerprint scores, and taking the top items, we obtain the recommendations made using the fingerprints. The quality of the fingerprint method is determined by its *precision*, the proportion of relevant items out of all the fingerprint recommendations. The following figure presents the relation between the fingerprint size (number of hashes used), and the quality of the recommendations.

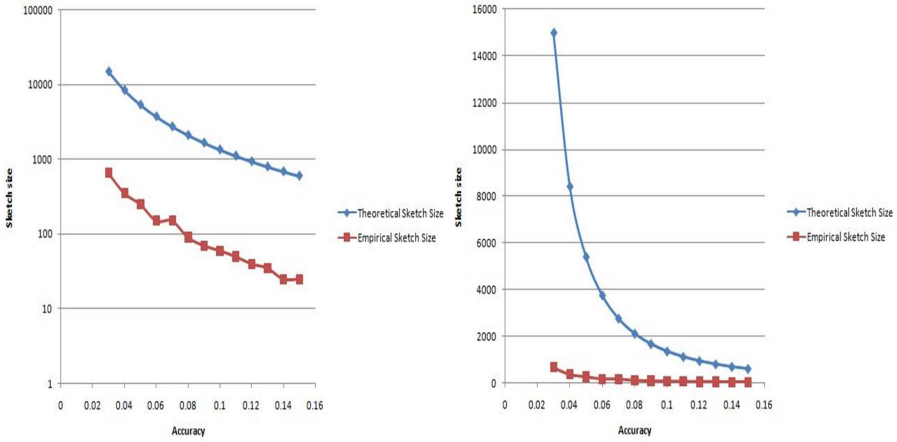


**Fig. 2.** Required sketch sizes for target accuracy, for confidence 0.9 (left - logarithmic scale )

Another important parameter is the *recall* of the method, the proportion of the relevant items that are covered. The recall is the proportion of relevant items (top 5% items under the true scores) covered by the top 5% of the items under the fingerprint scores. The recall values we measured range from 26% for 250 hashes to 33% for 1500 hashes. These results indicate that the quality of the recommendations is strongly related to the length of the fingerprint used. As seen in Figure 3, although longer fingerprints increase the quality, the quality improvement rate drops as more hashes are used. In some domains fingerprinting may allow the data to fit in RAM, rather than secondary storage (disks), and we suggest choosing the highest fingerprint length that allows the data to fit in memory, to maximize recommendation quality.

## 5 Related Work

We analyzed the famous Netflix dataset [8], a relatively recent CF domain. Early recommender systems include GroupLens [20] and Ringo [22]. Today's CF systems, as used by Amazon.com, MovieFinder.com and Launch.com face massive datasets. CF algorithms correlate human ratings to predict future preferences. There are many such correlations, such as the Pearson correlation used in [20] or the cosine similarity used in [9]. We focused on fingerprinting in massive CF systems. Similar works use fingerprints to approximate relations between strings. The work [13] presents a sketch for the $L_1$-difference, and [12] examines Hamming norm. This work extends [7, 6]. Both are purely theoretical, while this work includes theoretical analysis of different rank correlations and empirical analysis. While we use MD5 hashes for the empirical analysis, our theoretical results are based on MWIF hashes. MWIFs were treated in [10, 17]. We hope such

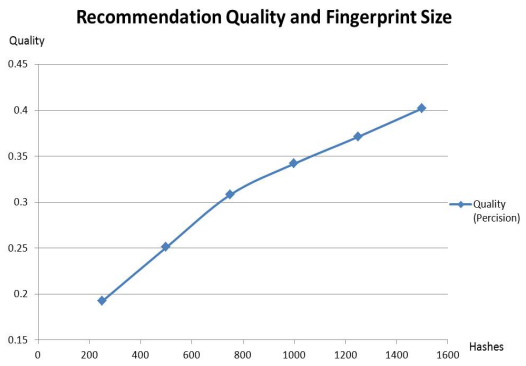


**Fig. 3.** Recommendation quality

techniques can be used to build fingerprints for various uses, such as collaborative filtering [7], trust and reputation aggregation [18, 5] and general preference aggregation and voting procedures [14].

Other techniques also concisely represent data relations. Our methods use the Locally Sensitive Hashing (LSH) [2] framework, but our analysis is based on assumptions that are specific to CF. Similar approaches are Random Projections [1] and Spectral Hashing [24]. Our methods are simple and efficient, and the empirical analysis shows they perform well on real CF datasets.

## 6 Conclusion

We suggest fingerprinting methods for CF systems, extending previous works to allow computing a family of rank correlations, including Spearman's Rho. We also provide empirical analysis of the suggested methods. Our results are based on *Collabriprint*, a complete fingerprinting based recommender system. Our results show that it is possible to use simple hash functions (rather than MWIFs) and short fingerprints to obtain high quality recommendations.

Several questions remain open for future research. First, we used a simple CF approach, and it would be interesting to see how the fingerprint size affects more sophisticated approaches. Also, it might be possible to create more sophisticated fingerprints to improve recommendation quality while still keeping the fingerprints small. Also, an even shorter fingerprint may be possible in certain restricted domains. Finally, other fingerprinting applications would be welcome.

## References

1. Dimitris Achlioptas. Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *JCSS*, 66, 2003.
2. Alexandr Andoni and Piotr Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. *Commun. ACM*, 51(1):117–122, 2008.

3. Y. Bachrach, N. Betzler, and P. Faliszewski. Probabilistic possible winner determination. *AAAI*, 38, 2010.
4. Y. Bachrach, E. Markakis, E. Resnick, A.D. Procaccia, J.S. Rosenschein, and A. Saberi. Approximating power indices: theoretical and empirical analysis. *Autonomous Agents and Multi-Agent Systems*, 20(2):105–122, 2010.
5. Y. Bachrach, A. Parnes, A.D. Procaccia, and J.S. Rosenschein. Gossip-based aggregation of trust in decentralized reputation systems. *Autonomous Agents and Multi-Agent Systems*, 19(2):153–172, 2009.
6. Yoram Bachrach, Ralf Herbrich, and Ely Porat. Sketching algorithms for approximating rank correlations in collaborative filtering systems. In *SPIRE 2009*, Saariselk, Finland, August 2009.
7. Yoram Bachrach, Ely Porat, and Jeffrey S. Rosenschein. Sketching techniques for collaborative filtering. In *IJCAI 2009*, Pasadena, California, July 2009.
8. Robert M. Bell and Yehuda Koren. Lessons from the netflix prize challenge. *SIGKDD Explor. Newsl.*, 9(2):75–79, December 2007.
9. J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of UAI-1998*, pages 43–52. Morgan Kaufmann, San Francisco, CA, 1998.
10. Andrei Z. Broder, Moses Charikar, Alan M. Frieze, and Michael Mitzenmacher. Min-wise independent permutations. *JCSS*, 60(3):630–659, 2000.
11. R. Clifford, K. Efremenko, E. Porat, and A. Rothschild. K-mismatch with don't cares. In *European conference on Algorithms*, pages 151–162, 2007.
12. Graham Cormode, Mayur Datar, Piotr Indyk, and S. Muthukrishnan. Comparing data streams using Hamming norms. *IEEE Trans. Knowl. Data Eng*, 15(3):529–540, 2003.
13. Joan Feigenbaum, Sampath Kannan, Martin Strauss, and Mahesh Viswanathan. An approximate L1-difference algorithm for massive data streams. *SIAM J. Comput*, 32(1):131–151, 2002.
14. E. Hemaspaandra, H. Spakowski, and J. Vogel. The complexity of Kemeny elections. *Theoretical Computer Science*, 349(3):382–391, 2005.
15. J.J. Higgins. *An introduction to modern nonparametric statistics*. Thomson Learning, 2004.
16. Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.
17. Piotr Indyk. A small approximately min-wise independent family of hash functions. *Journal of Algorithms*, 38(1):84–90, January 2001.
18. A. Jusang, R. Ismail, and C. Boyd. A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618–644, 2007.
19. Maurice G. Kendall. A new measure of rank correlation. *Biometrika*, 30:81–93, 1938.
20. Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstorm, and John Riedl. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of the ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, North Carolina, 1994. ACM.
21. Ronald L. Rivest. The md5 message-digest algorithm (rfc 1321).
22. Upendra Shardan and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *ACM CHI'95*, volume 1, pages 210–217, 1995.
23. C. Spearman. The proof and measurement of association between two things. by c. spearman, 1904. *The American journal of psychology*, 100(3-4):441–471, 1987.
24. Yair Weiss, Antonio Torralba, and Rob Fergus. Spectral hashing. In *Advances in Neural Processing Systems*, 2008.