# Audio Shingling for Measuring Musical Similarity *


## ABSTRACT

In this paper we present methods for identifying derivative works in large audio collections, that is works that contain audio passages that resemble passages from a source work, or set of source works. In our application, resemblance is approximate, we do not look for exact matches of the signal. This is because derivative works do not simply contain "samples" of the signal of an original work, but instead use one aspect of the source, such as a vocal passage, and remix it with new percussion and instrumental audio. Only a very small part of the source work might be used for the derivative work, so any method used to identify derivative must identify sources in a completely new context, this is called partial containment. Hence identification of derivative works is partial containment of approximately matching audio. Our solution uses audio shingling, a novel method based on techniques for near-duplicate elimination in document search. We describe the steps for constructing and comparing audio shingles, and for identifying partial containment using scalable hashing algorithm for nearest neighbour retrieval.



## General Terms

Music similarity, audio shingling, nearest neighbours in high dimensions


## 1. INTRODUCTION

This paper proposes a new way to measure audio similarity in very large databases using an algorithm we call audio shingles.

Our problem is interesting because today's on-line music libraries contain upward of 2 million songs. A brute force approach to find similar songs requires checking every frame of an audio file against every other frame, or $3.6B^2$ distance calculations. This is clearly impractical. Conventional approaches to organizing the search space, such as different forms of trees, fail due to the curse of dimensionality; Our feature vector contains 1200 dimensions. In such a high-dimensional space, nearest neighbor calculations are difficult.

We're interested in audio similarity because we want to help users find the music they are interested in. There are two practical needs driving this work. First, user's often have a playlist and want to move it to a new system. We want to be able to offer the user a close match if we don't have the exact song title. Second, and perhaps more importantly, commercial success in these days of large music catalogs is based on finding the music that people want to listen to. This is driven by a recommendation system, which depends on user's rating data. A recommendation system will perform much better if we can propagate a user's rating to other recordings of the same song.

### 1.1 Audio Similarity

It is difficult to define similarity and even more difficult to score results. For the purposes of this work, we say two songs are similar if one is a derivative of another. Derivative works do not simply contain "samples" of the signal of an original work, but instead use part of a vocal track and remix it with new percussion and bass tracks. Furthermore, only a small part of the source work might be used for the derivative work, so any method used to identify derivative works must be able to identify a small amount of material in a completely new context; this is called partial containment. Hence identification of derivative works requires determining partial containment of approximately matching audio.

Examples of derivative works in large commercial music collections are titles released in different formats, e.g. single, extended play single, radio edit, summary clip, album version, DJ version for clubs, re-mixes by different producers, studio session out-takes, multiple takes from a recording session and multiple performances in different sessions / concerts. In this paper, we present methods for identifying songs that are related by some degree of duplication of material, even when the degree of overlap between the source and the derivative work is relatively small, on the order of a few seconds, and then it is approximate, i.e. not containing identical material from the source. For purposes of testing in this work, we say two songs are similar if the words in their title overlap.

*(Produces the permission block, and copyright information)

Our work is different from the work that has been done on audio fingerprinting [18][19][17]. With fingerprinting users want to find the name of a recording given a (lousy) sample of the audio. The secret sauce that makes fingerprinting work is based on defining robust characteristics of the signal that lend the song its distinctive character, and are not harmed by difficult communications channels (i.e. a noisy bar and a cell phone). Often these systems assume that some portion of the audio is an exact match–this is necessary so they can reduce the search space. We do not expect to see a exact match anywhere and we are interested in ranking the songs that are similar to each other.

## 1.2 Locality Sensitive Hashing

Our audio work is based on an important new web algorithm known as shingles and a randomized algorithm known as locality-sensitive hashing (LSH). Shingles are a popular way to detect duplicate web pages and to look for copies of images. Shingles are one way to determine if a new web page discovered by a web crawl is already in the database. Text shingles use a feature vector consisting of word histograms to represent different portions of a document. Shingling's efficiency at solving the duplicate problem is due to an algorithm known as a locality-sensitive hash (LSH). In a normal hash, one set of bits (e.g. a string) is transformed into another. A normal hash is designed so that input strings that are close together are mapped to very different locations in the output space. This allows the string-matching problem to be greatly sped up because it's rare that two strings will have the same hash.

LSH, instead, does exactly the opposite; two patterns that are close together are hashed to locations that are close together. Each hash produces an approximate result since there is always a chance that two nearby points will end up in two different hash buckets. Thus, we gain arbitrarily-high precision by performing multiple LSH mappings, each from a different random direction, and noting which database frames appear multiple times in the same hash bucket as our query. Each hash can be as simple as a random projection of the original high-dimensional data onto a subspace of the original dimensions. Furthermore the dimensionality of the input space is often greatly expanded by representing each dimension as a unary code. In an unary code, the value X of a dimension is represented as a string of X ones followed by N-X zeros to give a N-dimensional binary vector. This binary representation of the data is important because the output subspace is now a binary vector and is representable as a word in a computer's memory.

## 1.3 Data Set

We performed our experiments on the complete recordings of two artists, Madonna and Miles Davis. These two artists were chosen because they both have extensive back catalogs and their music is available electronically. Each recording has a unique 20-digit unique identifier (UID) that is used to locate metadata such as artist, title, album and song length. We obtained exact copies of each commercially distributed recording in a *lossless* format from the Yahoo YMU warehouse (80GBytes of data) and performed our feature extraction directly on the 44.1kHz PCM representation. Our experiment catalogue consists of 306 separate Madonna recordings and 1712 separate Miles Davis recordings. The

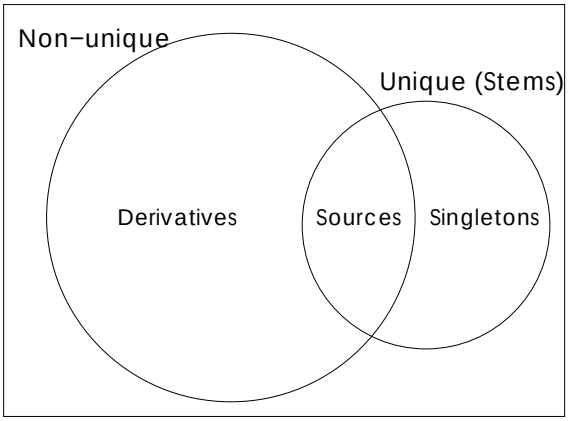


**Figure 1: Venn diagram showing the distribution of titles in the database. The unique titles are the set of stems, these are divided between those that are singletons and those that have related works in the database by title stem. Of the non-unique set, some are the source works and the rest are the derived works.**

total duration of audio was 222 hours 26 minutes and 14 seconds.

On inspecting the catalogue, it is immediately apparent that many separate recordings share all, or part, of their title string. And that partial matches are anchored sub-strings with a parenthesis delimiting the title stem from any extended title information. The following table illustrates the related titles for the Madonna work *Nothing Fails*.

| Duration | Title |
|---|---|
| 3m55s | Nothing Fails (Nevins Mix) |
| 7m27s | Nothing Fails (Jackie's In Love In The Club Mix) |
| 7m48s | Nothing Fails (Nevins Global Dub) |
| 7m32s | Nothing Fails (Tracy Young's Underground Mix) |
| 6m49s | Nothing Fails (Nevins Big Room Rock Mix) |
| 8m28s | Nothing Fails (Peter Rauhofer's Classic House Mix) |
| 4m49s | Nothing Fails |
| 3m48s | Nothing Fails (Radio Edit) |
| 4m0s | Nothing Fails (Radio Remix) |
| 4m49s | Nothing Fails |

One of the advantages of working with intra-artist derivative works identification is that a collection of ground truth can be gathered by matching stemmed song titles. To stem the titles, we first removed any puncuation, such as quotation marks, and truncated each title up to the first parenthesis if present, else no truncation occured. Any leading or trailing whitespace after these transformations was also removed. For example, all of the titles in the table above were transformed by the stemming to the string *"Nothing Fails"*.

Once the titles in the database were stemmed, we gathered statistics on title use within each artists' collection of songs. This is summarized in Figures 1, 2 and in the following table:

| Artist | Tracks | Stems | Sources | Derivatives |
|---|---|---|---|---|
| Madonna | 306 | 142 | 82 | 164 |
| Miles Davis | 1712 | 540 | 348 | 1172 |

There were 306 different Madonna recordings in the database

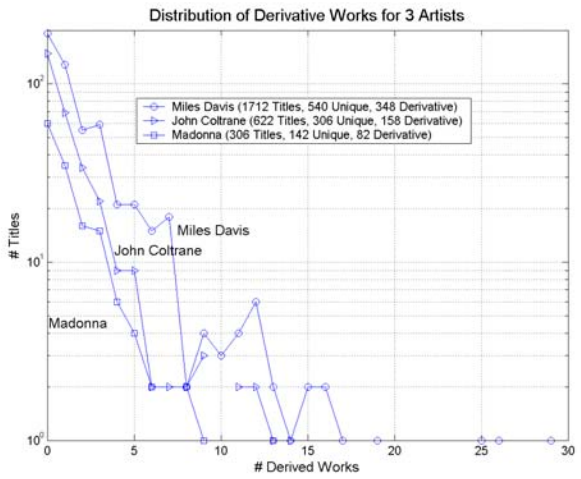


**Figure 2: Distribution of titles in the database showing the number of derivative works on the $x$ axis and number of unique titles sharing cardinality of derived works on the $y$ axis. More than half the titles of each of the three artists shown had derivative works such as alternate recordings, re-edits and remixes.**

with 142 had unique title stems, 82 of which had derivative versions (58%), giving a total of 164 derivative works. Similarly, there were 1712 different Miles Davis recordings, with 540 unique title stems, of these 348 had derivative versions (64%) giving a total of 1172 derivative works.

We used 20 Madonna songs with deriative works as our test set. From the set of songs with the same title stem, a "source" song was selected as being the historically earliest version of the song in the database. The number of relevant matches for the set of 20 such source queries (not including the queries themselves) is 76 songs of 2018.

## 1.4 Contributions

This work makes 3 contributions to the literature. First, we confirm an earlier result [23] showing that matched filters, and thus Euclidean distance in a feature space, are an effective way to measure song similarity. In this paper we show results for a much larger database of songs. Second, we investigated and describe several variations of our basic features that perform better on this large-scale task. Third, we introduce LSH as a nearly exact means of finding nearest neighbors in a large audio space. We also report on the distribution of audio shingles.

# 2. PREVIOUS WORK

To date, a range of feature-based techniques have been proposed for describing and finding musical matches from a collection of audio. Figure 3 shows the range of options. Fingerprinting [9] finds the most salient portions of the musical signal and uses detailed models of the signal to look for exact matches. At the other end of the specificity scale, genre-recognition [22], artist recognition [4], musical key identification [12], and speaker identification [13] use much more general models such as probability densities of acoustic features

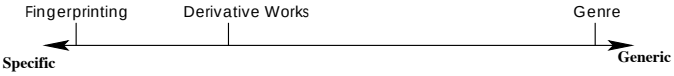


**Figure 3: Specificity of derivative works identification. The most specific queries are on the left of the figure and the most generic on the right. Derivative works identification, as described in this work, falls in between.**

approximated by Gaussian Mixture Models. These so-called bag-of-feature models ignore the temporal ordering inherent in the signal and, therefore, are not able to identify specific content within a musical work such as a given melody or section of a song.

Instead, we propose retrieval of musical extracts from large collections by approximate matching of feature sequences. In this problem, musical passages are equivalent if a human listener would judge them similar, even if they are acoustically distinct. An example application is the retrieval of "cover" versions of a song, whether they are by the same or different artists, instrumental versions or different arrangements. This type of problem is important to music download and archiving services, for example, because duplicate popular songs (such as "Yesterday") can have many hundreds of entries in a large database, this clutters retrieval lists with unnecessary repetitions of different versions of the same title. The same problem is encountered with the numerous re-mixes, re-edits and re-releases of the most popular artists' tracks. The problem is analogous to near-duplicate elimination in text document and image archives and has many interesting analogues in the audio domain [3] [10].

These applications, as well as many others, require algorithms that are robust to differences in the lyrics, instrumentation, tempo, rhythm, chord voicing and so forth, so we propose to explore features that are invariant to various combinations of these [1][2]. We are interested in all types of music, including classical works, popular music tracks and electronic music. The applications of such similarity methods are far-reaching, and have immediate relevance to music browsing, musicology, structure identification and musical creativity [16][5].

Inherent in this problem is the need to measure distances in a perceptually relevant fashion and quickly find similar matches without an exhaustive search through the entire database. Existing Gaussian Mixture Model methods for computing audio similarity do not scale to large databases of millions of songs due to the computation required in pairwise comparison of models using a suitable distance function such as Earth Movers Distance (EMD) [11]. Likewise, high-dimensional feature representations are susceptible to the curse of dimensionality that leads to inefficient (linear time) search algorithms. We will fail in large databases if we need to look at every signal to decide which are closest.

Recent work shows that audio features can be efficiently retrieved using locality-specific hashes (LSH) which have sub-linear time complexity in the size of the database. This is a key requirement for audio retrieval systems to scale to searching in catalogues consisting of many millions of en-

tries. These methods have already found applicability in image-retrieval problems [3]. LSH solves approximate nearest neighbour retrieval in high dimensions by eliminating the curse of dimensionality [8][6][7].

The features used to describe the signal are critical. LSH is only appropriate when the signal can be represented by a point in a fixed-dimensional metric space with a simple norm (such as L2). For example, methods that compare sequences of different lengths, such as dynamic time warping, are not easy to implement using LSH. Other models fail this metric because the distance measure is not simple. This includes Gaussian mixture models, hidden Markov models, and distance metrics based on dynamic time warping. It is shown in [23] that LSH is theoretically able to solve the audio sequence search problem accurately, and in sub-linear time, when the similarity measure is a convolution of sequences of audio features which is provides an L2 norm.

Our previous work has shown that matched filters, and therefore Euclidean distance, using chromagram and cepstral features performs well for measuring the similarity of passages within songs [23]. The current work applies these methods to a new problem, grouping of derived works and source works in a large commercial database using an efficient implementation based on LSH.

# 3. AUDIO PROCESSING

One aspect of our work that sets it apart from previous work in audio fingerprinting is the use of perceptually-motivated features, and then representing the audio in a time series of feature vectors called audio shingles. The goal is to represent the audio signal in a feature space so that items that are perceptually close will end up being neighbours in the feature space. We examined two contrasting audio features. The first was log frequency cepstral coefficients (LFCC) and the second was the chromagram (CHROM).

The LFCC and CHROM feature used in this paper both measure aspects of the music that are perceptually relevant to our similarity task. The LFCC representation, a simplification of the Mel-frequency cepstral coefficiients (MFCC) used in speech recognition work [14], measures the shape of the overall spectrum. It gives us a 20-dimensional representation of the timbre. The CHROM feature, on the other hand, is a measure of the musical notes found in each frame of music. The entire spectrum is collapsed into a probabilistic measure of the 12 musical notes found at that point in time.

We investigated the entropy of the distributions as a way to understand how specific each representation is. In a one-note tune the chromogram has one non-zero entry and thus a very low entropy. With noise, the spectrum and chromogram are flat and thus the entropy is maximum.

## 3.1 Feature Extraction

The uncompressed 44.1kHz PCM audio signals are first segmented into length 372ms frames overlapped with a hop size of 100ms. A hop size of 100ms was chosen to trade off temporal acuity against time and space complexity for the search. Previous work indicates that, even at the signal level, the spectrum is sufficiently correlated in time that small shifts in frame alignment lead to small changes in feature values.

At the first stage we wanted to reduce the sensitivity of our feature to only those parts of the spectrum that human hearing is sensitive. To this end, informed by previous research into human hearing and auditory filterbanks [14], we used a constant-Q logarithmic frequency filterbank to extract power coefficients in non-overlapping logarithmic frequency bands, the bandwidth is proporionally higher for higher frequency components. This makes a resulting similarity measure less sensitive to changes in individual high-frqeuency components but more sensitive to changes in individual low-frequency components.

We derived two features from the constant-Q spectrum transform. LFCC features are extracted using a 16th-octave filterbank and chromagram features are extracted with a 12th-octave filterbank. In both cases the filterbank extended from 62.5Hz to 8kHz. The filterbank was normalized such that the sum of the logarithmic band powers equalled the total power.

To extract the LFCC coefficients we used a discrete cosine transform (DCT) retaining the first 20 of 114 spectral coefficients. To extract CHROM features we summed the energy in logarithmic bands corresponding to octave multiples of 12 reference pitch classes corresponding to the set $\{C, C\#, D, ..., A\#, B\}$.

## 3.2 Audio Shingles

We create a shingle by concatenating 100 frames of 12-dimensional chromagram features into a single 1200 dimensional vector. Much like the original work on shingles [3], we advance a pointer by one frame time, 100ms, and then calculate a new shingle. Unlike text shingles, which are word histograms, our shingles are time-varying vectors. To make the shingles invariant to energy level we normalized the shingle vectors to unit length.

we used the vector dot product to compute the similarity between a pair of shingles This can be computed efficiently for audio shingles using convolution, which is proportional to the L2 (Euclidean) distance between them. This is shown as follows.

Consider the L2 norm at one time instant, $t$, between the current signal, $x(t)$, and a query, $y(t)$.

$$D(t) = \sum_s [x(s) - y(t-s)]^2 \quad (1)$$

$$D(t) = \sum_s [x(s)^2 - 2x(s)y(t-s) + y(t-s)^2] \quad (2)$$

$$D(t) = \sum_s x(s)^2 + \sum_s y(t-s)^2 - 2\sum_s x(s)y(t-s) \quad (3)$$

If the energy in the signal and the query are normalized, and independent of time and query, then the last term is proportional to a matched filter implementation of the query

operator. The matched filter is the optimal linear signal detector and is amenable to an implementation based on LSH as we discuss below.

### 3.3 Similarity Measurement

Our similarity measurement is performed in two stages. We first search for the $N$ audio shingles in our database that are closest to each query song. Given these nearest-neighbor matches, found using brute force or LSH, we look at the top-N shingle matches for a pair of songs and compute the similarity by averaging these smallest-N distance scores to find the similarity between the two songs. Thus a short fragment that is contained in another song will cause the similarity measure to be small and indicate a close match.

The distance function between two songs, $S_i^a = S(X_a, w)$ and $S_j^b = S(X_b, w)$, is the mean distance of the k-nearest neighbour pairs of shingles between them. In our experiments we used 20 query songs taken from the 2018 database. Queries were selected that had mutliple derivative versions in the database.

The length of a vector describing a shingle, and thus the length of the underlying matched filter is fixed in time. On one hand we want the shingles to be long so they are more specific. On the other hand, they need to be short so that variations in tempo do not build up enough so that the simple matched filter fails.

The effect of matched-window size on matching performance is shown in Figures 4 and 5. These figures show the cross-chromagram for two Madonna songs, one a derivative of the other. There is slightly crooked white diagonal line from upper left to lower right showing that the two songs are roughly aligned in time. (The fact that the line is not straight is caused by tempo variations between the two songs.) For each column of the matrix, a red "x" is shown at the point where the chromagram distance is smallest in the other song. Only the top 100 matches are shown in these figures. Figure 4 shows the cross-chromagram for a 1-second long window, and there are many false matches. Figure 5 shows the matches for the same songs, but with 10-second long windows. There are fewer false positives. A previous study showed that matched filters of length 3s were optimal [23]. The white bars around the image are places where the cross-chromagram is not computed because the energy is too low.

We used a new version of LSH based on p-stable distributions [6]. With a p-stable distribution, vector sums of random variables from a p-stable distribution still have the same probability distribution. We form a number of dot products between the database entries and random variables from the p-stable distribution. Each of these dot products forms a projection onto the real axis. We can then divide up the real axis into buckets and form a hash that is locality specific–points that are close together in the input space will be close together after projection onto the real axis.

## 4. EXPERIMENTAL RESULTS

We first present audio similarity results with audio shingles using convolution distance. Then we present results using an efficient implementation based on hashing.

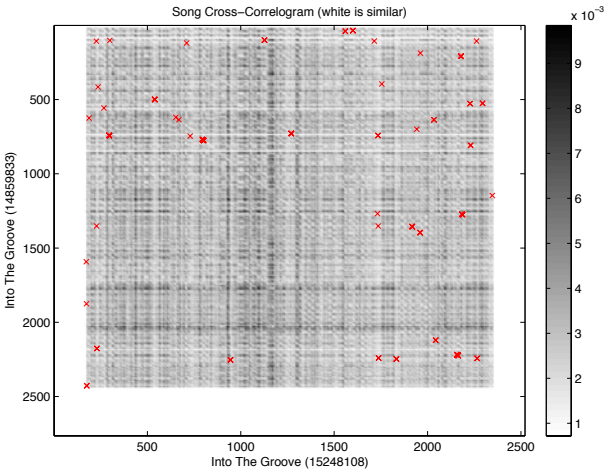


**Figure 4: Cross chromagrams for two Madonna songs showing distances for a source and a derived work calculated with 1s windows as a function of time. The best matches for each column are shown with a red 'x'.**

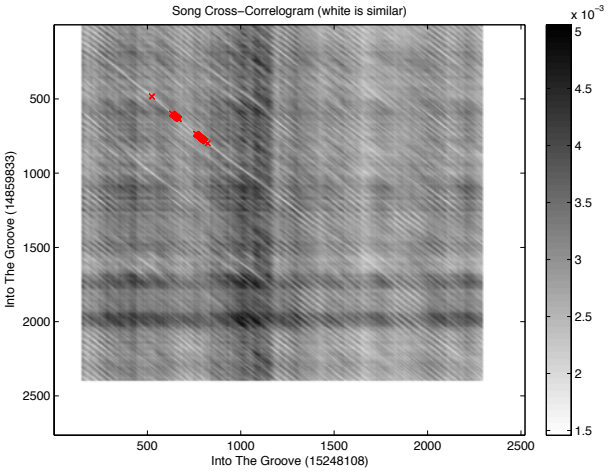


**Figure 5: Cross chromagrams for two Madonna songs showing distances for a source and a derived work calculated with 10s windows as a function of time. The best matches for each column are shown with a red 'x'.**

We tried several different methods to remove noisy signals from our database. Our first experiment didn't work because near-silent regions of some song were the best matches. This was because the features were constructed to be invariant to changes in energy, but for low energy levels the estimates of the feature were noisy thereby generating nearest neighbour matches from a random distribution of features.

To compensate, we ran a second experiment that included only those shingles with energy, integrated over the match window, above the mean energy for each track. The performance improved significantly, however, we noticed that many of the matches were short clips containing mostly recording studio ambience, near silence with tape hiss, with a small amount of dialogue. The reason for the poor performance was that near-silence was dominating the entire track, so the mean energy was very low, so these matches suffered from a similar effect to our first experiment.

In our final experiment, reported here, silence was first removed using an absolute threshold and relative low-energy shingles were removed using the method described above. The precision improved significantly. These are the results presented here.

## 4.1 Experiment 1: Exact Nearest Neighbours

We used 100-frame shingles (10s) for the experiment with a hop size of one frame ($0.1s$). For each song in the detabase we found the 100 Nearest Neighbours, that is the pairs of query/database points with the 100 smallest distances. The average of these distances was taken as the similarity between the query song and the database song. This yielded a list of $N$ similarity ratings between the query song and the $N$ database songs. This operation was performed for all of 20 query songs.

Evaluation took place by counting the number of true positives and false positives at each level of recall standardized into 10th-percentiles. We calculated confidence intervals using the standard deviation of the precisions at each 10th-percentile interval and dividing by the square root of the number of queries.

Figure 6 shows the precision recall performance for four different feature arrangements. The lowest ranking features were LFCC used on its own and CHROM with a 2kHz cutoff frequency. CHROM with 8kHz cutoff performed the best out of the individual feature spaces.

We also tried a joint feature space consisting of both CHROM and LFCC features. Here, the song similarity measure is a weighted average of the chromagram and lfcc features. Results are shown for 0.9*CHROM + 0.1*LFCC; an empirically determined mixture of the distances. The improvement in retrieval can be accounted for that the false negative rate is reduced but not the false positive rate using the joint feature space. CHROM and LFCC encode qualitatively different aspects of the songs. CHROM features encode the harmony and pitch content, and LFCC features encode the timbral content. We were surprised that the weighted average performed better and we are investigating the reason.

To see how the rerieval performance scaled, we repeated the

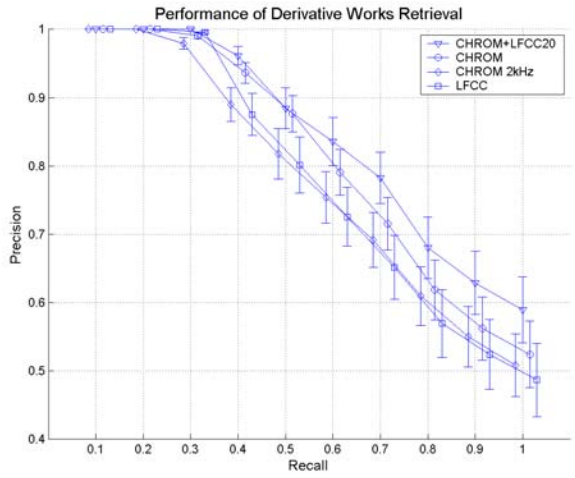


**Figure 6: Results of k-NN audio shingle retrieval of derivative works in a 306-song subset of the 2018 song database.**

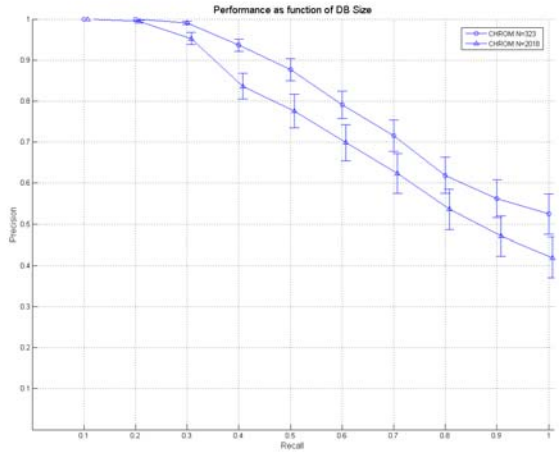


**Figure 7: Comparative performance for database of 306 songs and full database of 2018 songs.**

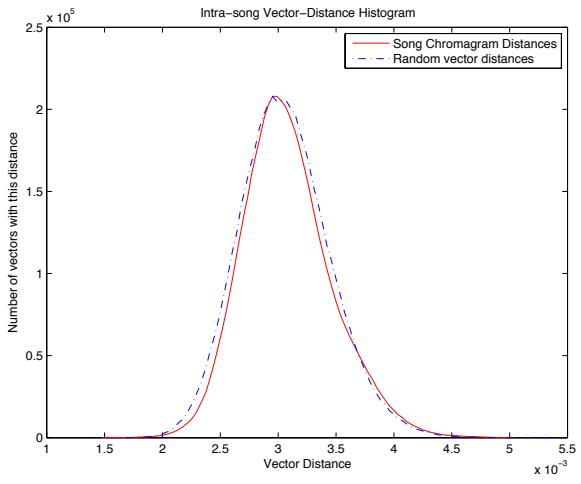


**Figure 8: Intra-song Vector Distance Histograms. Comparing the distance between 100-frame chromagram features and (solid line) and two Gaussian random vectors (dashed line.)**

experiments above using the 2018 song database (See Figure 7.) We note that there was a 10% drop in precision for the larger database at recall rates $> 40\%$.

Our results are an error rate of 34% at a recall of 80%. The time complexity of the exact approach is average case $|Q| \times |\bar{S}| \times d \times w \times \mathbf{O(N)}$ in the number of songs in the database, $N$, the number of query shingles, $|Q|$, the average number of shingles per song,$|\bar{S}|$, the feature dimensionality, $d$, and the length of the shingles, $w$.

For the twenty queries used in our experiments, using chromagram features, this results in approximately $2018 * 20 * 3000 * 3000 * 12 * 100 = 4.3 * 10^{14}$ multiply-accumulate operations. Computation took approximately 7 hours per song, average query time.

## 4.2 Experiment 2: Locality Sensitive Hashing Search

The application requires approximate shingle matching. We do this by solving the nearest neighbours problem in high dimensions using locality sensitive hashing.

Our data is not like the data usually used for LSH–our data is much more random. Often LSH is applied to problems where there are one or two duplicates in close proximity, and then a wide gap before reaching the rest of the points. This is not true in our case.

One consequence of working in random high-dimensional spaces (1200D in our case) is that all points tend to be on the shell of the sphere, and are equidistance from each other. Figure 8 shows a plot of the inter-point distances between chromagram shingles and random Gaussian-distributed vectors. The distance histograms, after scaling, are nearly identical. This equidistance behavior, and the exponential growth of the distance histogram means that it is hard to pick the right radius for the nearest-neighbor calculation for this application of LSH.

The method for LSH is slightly different than the pair-wise method in that it returns the indices of all shingles that are within a specified radius of each query point. As the radius increases, the false positive and true positive counts also increase. The output of the LSH program is a list of all retrieved points for each query point that were within the given radius. We identified the songs to which each retrieved point belonged. The average distance for the 100 nearest points to the query song was used as a similarity score.

For the LSH method, we note that retrieval performance for derivative works was similar to the pair-wise method. However, the time for computation was substantially lower than for pair-wise computation.

# 5. CONCLUSIONS

Audio shingles are an efficient means of identifying similar and derivative musical works. We discovered that near-silence and low-energy suppression is important for good results. The brute-force, pair-wise method gave good results, but it is very slow. LSH is at least an order of magnitude faster, and our matched-filter formulation is a success.

A current limitation of the LSH method is that it is memory intensive. For longer audio-shingles the space requirement for 2000 songs was close to 4GB. If thehost system swapst hen temporal performance drops by several orders of magnitude, sometimes this has the effect of being slower than pair-wise computation.

# 6. REFERENCES

[1] Mark A. Bartsch and Gregory H. Wakefield. To Catch a Chorus: Using Chroma-Based Representations for Audio Thumbnailing. in *Proc. WASPAA*, 2001.

[2] J. P. Bello, and J. A. Pickens. A Robust Mid-level Representation for Harmonic Content in Music Signals. *Proceedings of the 6th International Conference on Music Information Retrieval (ISMIR-05),* London, UK. September 2005.

[3] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. In Proceedings of WWW6 '97, pages 391–404, Elsevier Science, April 1997.

[4] D. Ellis, B. Whitman, A. Berenzweig, S. Lawrence. The Quest for Ground Truth in Musical Artist Similarity. *Proc. ISMIR-02*, pp. 170–177, Paris, October 2002.

[5] Jonathan Foote. Visualizing music and audio using self-similarity. in *ACM Multimedia (1)*, 1999, pp. 77–80.

[6] M. Datar, P. Indyk, N. Immorlica and V. Mirrokni. Locality-Sensitive Hashing Scheme Based on p-Stable Distributions, In *Proceedings of the Symposium on Computational Geometry*, 2004

[7] Locality-sensitive hashing using stable distributions, in *Nearest Neighbor Methods in Learning and Vision: Theory and Practice* , by T. Darrell and P. Indyk and G. Shakhnarovich (eds.), MIT Press, to appear.

[8] Aristides Gionis, Piotr Indyk and Rajeev Motwani. Similarity Search in High Dimensions via Hashing. *The VLDB Journal,* pp. 518–529, 1999.

[9] J. Herre, E. Allamanche, O. Hellmuth, T. Kastner. Robust identification/fingerprinting of audio signals using spectral flatness features. *Journal of the Acoustical Society of America*, Volume 111, Issue 5, pp. 2417–2417, 2002.

[10] Yan Ke, Rahul Sukthankar, Larry Huston. An efficient near-duplicate and sub-image retrieval system. *ACM Multimedia,* 2004: 869–876.

[11] B. Logan and S. Chu. Music Summarization Using Key Phrases. *In Proc.IEEE ICASSP*, Turkey, 2000.

[12] S. Pauws. Musical Key Extraction from Audio. *In Proc.ISMIR*, Barcelona, 2004.

[13] Douglas A. Reynolds. Speaker identification and verification using Gaussian mimxture speaker models. *Speech Commun.,* 17 (1–2):91–108, 1995.

[14] This reference has been removed to preserve anonymity.

[15] Eric Scheirer and Malcolm Slaney. Construction and evaluation of a robust multifeatures speech/music discriminator. *IEEE Transactions on Acoustics, Speech, and Signal Processing* (ICASSP'97), 1997, pp. 1331–1334.

[16] D. Schwartz. The Caterpillar System for Concatenative Sound Synthesis. *Proc. DAFx*, London, 2003.

[17] Matthew Miller, Manuel Rodriguez and Ingemar Cox. Audio Fingerprinting: Nearest Neighbour Search in High Dimensional Binary Spaces. Multimedia Signal Processing, 2002 IEEE Workshop on , 2002

[18] Jaap Haitsma, Ton Kalker. A Highly Robust Audio Fingerprinting System, Proc. *ISMIR*, Paris, 2002.

[19] P. Cano and E. Batlle and T. Kalker and J. Haitsma, A review of algorithms for audio fingerprinting. In International Workshop on Multimedia Signal Processing, US Virgin Islands, December 2002.

[20] Avery Li-Chun Wang, Julius O. Smith, III. System and methods for recognizing sound and music signals in high noise and distortion. United States Patent 6990453, 2006

[21] Norbert Wiener. *Extrapolation, Interpolation, and Smoothing of Stationary Time Series.* New York: Wiley, 1949.

[22] G. Tzanetakis and P. Cook. Musical genre classification of audio signals. *IEEE Transactions on Speech and Audio Processing,* 10(5):293–302, 2002.

[23] This reference has been ommitted for anonymous submission.