Early MFCC And HPCP Fusion for Robust Cover Song Identification

Christopher J. Tralie
Duke University Department of Electrical and Computer Engineering
ctralie@alumni.princeton.edu

Abstract

While most schemes for automatic cover song identification have focused on note-based features such as HPCP and chord profiles, a few recent papers surprisingly showed that local self-similarities of MFCC-based features also have classification power for this task. Since MFCC and HPCP capture complementary information, we design an unsupervised algorithm that combines normalized, beat-synchronous blocks of these features using cross-similarity fusion before attempting to locally align a pair of songs. As an added bonus, our scheme naturally incorporates structural information in each song to fill in alignment gaps where both feature sets fail. We show a striking jump in performance over MFCC and HPCP alone, achieving a state of the art mean reciprocal rank of 0.87 on the Covers80 dataset. We also introduce a new medium-sized hand designed benchmark dataset called "Covers 1000," which consists of 395 cliques of cover songs for a total of 1000 songs, and we show that our algorithm achieves an MRR of 0.9 on this dataset for the first correctly identified song in a clique. We provide the precomputed HPCP and MFCC features, as well as beat intervals, for all songs in the Covers 1000 dataset for use in further research.

1 Introduction

A "cover song" is a different version of the same song, usually performed by a different artist, and often with different instruments, recording settings, mixing/balance, tempo, and key. To sidestep a rigorous definition, like others, we evaluate algorithms on a set of songs that have been labeled as covers of each other, and we declare success when our algorithm recognizes clusters of songs which have been deemed covers of each other. In fact, this problem is more of a "high level music similarity" task beyond exact recording retrieval, making the problem intrinsically more difficult than traditional audio fingerprinting [16, 31, 30].

Most work on automatic cover song identification to date has focused on estimating and matching note-based features such as chord estimates [1], chroma [8, 11], harmonic pitch class profiles (HPCP) [15, 23, 25], 2D Fourier magnitude coefficients to approximate these features [12, 19]. This is natural, since regardless of all of the transformations that can happen between versions, note sequences should be preserved up to transpositions. Problems occur, however, when note sequences are not the defining characteristic of a musical expression. This is common in hip hop, for example, such as the song "Tricky" in the "Covers 80 Dataset" ([10], Section 4.1) performed by Run D.M.C. and The Beastie Boys. There are also songs which are entirely percussive, such as the 8 covers of Frank Zappa's song "The Black Page" that we present in Section 4.3. Moreover, even for song pairs with strong harmonic content, there may be sections with drum solos, un-pitched spoken words, or other musical statements on which pitch-based features fail. However, the issue with features complementary to Chroma, such as Mel-Frequency Cepstral Coefficients (MFCCs), is that they are highly sensitive to instrument and balance changes. In spite of this, some recent works have shown that MFCC-based features can also be used in cover song identification [5, 29]. Particularly, if the *relative* changes of MFCC are captured, as in [29], performance is still reasonable.

Naturally, then, rather than relying on a single feature in isolation, recent works have shown the benefits of feature fusion after song comparisons have been made with each feature set alone. For instance, aggregating ranks from individual features can improve results [20, 21]. Other works show the advantage of using all pairwise similarities computed with different features [6], using a cross-diffusion process known as "similarity

network fusion" (SNF) ([32, 33], Section 3.1) to come up with a consensus similarity score between all pairs of songs in a corpus.

In this work, we develop a similarity network-based early fusion technique which achieves state of the art results by combining complementary HPCP, MFCC, and self-similarity MFCCs (SSM MFCCs). Unlike [6], we apply SNF *before alignment* in the space of features. This fusion technique incorporates both cross-similarity between two different songs and self-similarity of each song, so it is able to combine information about matching sections between songs and structural elements within each song. We also apply late fusion on similarity scores between a network of songs to further boost the results. While state of the art supervised techniques on the popular "Covers 80" benchmark dataset yield a mean reciprocal rank (MRR) of 0.625 [6][1], our completely unsupervised technique achieves a MRR of 0.87 (Section 4.1). We also introduce our own dataset consisting of 395 cliques of songs, which we call "Covers1000" (Section 4.2), and on which we report an MRR of 0.9 with our best fusion technique.

2 Beat-Synchronous Blocked Features

In this section, we describe three complementary features which we later fuse together in Section 3. One concept we rely on in our feature design is "block-windowing," which was described in [29]. The idea is that when a contiguous set of features in time are stacked into one large vector, they give more information about the local dynamics of a song. This is related to the concept of a delay embedding in dynamical systems [17]. Having many blocks across the song also allows us to control for drift by normalizing within each block. To control for tempo differences, we compute our blocks synchronized with beats, which is a common preprocessing step [8, 11, 29]. We use the simple dynamic programming approach of [9], since it allows the specification of a tempo bias. As in [11] and [29], we bias the beat tracker at 3 different tempo levels (60, 120, 180bpm), and we compare all pairs of tempo levels, for up to 9 unique combinations, since the beat tracker may choose an arbitrary level of subdivision in a rhythm hierarchy. Once the beat onsets are computed, we form a block for every contiguous group of B beat intervals, as in [29].

2.1 HPCP Features

One proven set of pitch-based features for cover song identification are "harmonic pitch class profiles" (HPCPs) [14] [2]. Following [26], we compute a stacked delay embedding of the HPCP features within B beats, with two HPCP windows per beat, for a total of $2B$ windows per block. This has an advantage over other works which do not use a delay, as it gives more context in time, and it is consistent with the block/windowing framework. To normalize for key transpositions, we need to determine an "optimal transposition index" (OTI) between two songs ([24]). Given the average HPCP vector $X \in \mathbb{R}^{+12}$ from song A and the average HPCP vector $Y \in \mathbb{R}^{+12}$ from song B, we compute the correlation $X^T Y$ over all 12 half-step transpositions of the original HPCP features in the block, and we use the transposition that leads to the maximum correlation. Then, we compute cosine distance between all pairs of HPCP blocks between the two songs.

2.2 MFCCs / MFCC Self-Similarity Matrices (SSMs)

In addition to HPCP features, we compute exponentially liftered MFCCs in beat-synchronous blocks. We take the MFCC window size to be 0.5 seconds, and we advance the window intervals evenly from the beginning of the block to the end of a block with a *hop size* of 512 samples. At a sample rate of 44100 Hz, this leads to a window size of 22050 and an overlap of roughly 97.5% between windows (Section 2.2). Longer windows have been shown to increase robustness of SSM matching in [29] and audio fingerprinting [16], which justifies this choice. To allow direct comparisons between different blocks, we interpolate to 400 MFCCs per block, and we perform Z-normalization (as in [29]) to control for loudness and drift, which we found to be an essential step.

[1] This technique scored the best in MIREX 2016

[2] To ensure we have a state of the art implementation, we use the Essentia library to compute HPCPs [4].

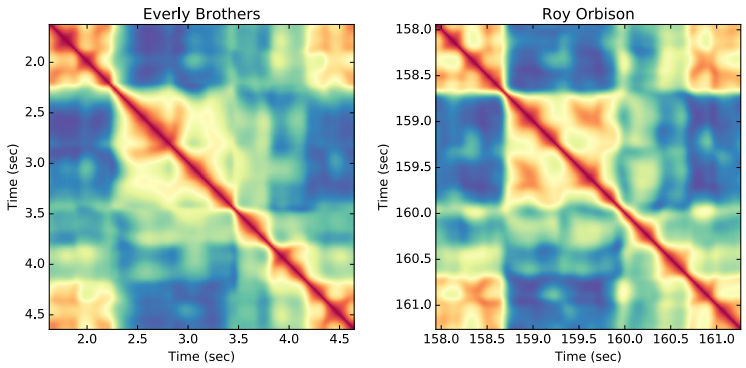

Figure 1: Example 8-beat Z-normalized MFCC SSMs blocks in correspondence between cover versions. A block from "Claudette" by the Everly Brothers and Roy Orbison. The pattern in this block is Guitar + "Oooh ooh Claudette" + Guitar.

In addition to block-synchronized and normalized raw MFCCs, we also compute self-similarity matrices (SSMs) of the Z-normalized MFCCs within each block, as in [29], leading to a sequence of SSMs for each song. That is, unlike [2], who compares SSMs between entire songs, we compare SSMs summarizing blocks of audio on the order of tens of beats, as recommended by [29]. For each beat-synchronous block, we create a Euclidean SSM between all MFCC windows in that block. As with the raw MFCCs, to allow comparisons between blocks, we resize each SSM to a common image dimension $d \times d$. Figure 1 shows two examples of MFCC SSM blocks with 8 beats and 500 windows per block which were matched between a song and its cover in the Covers80 dataset. Although the underlying sounds are quite different (male to female, different instruments and balance), the SSMs look similar. [29] argue that this is why, counter to prior intuition, it is possible to use MFCCs in cover songs.

2.3 Cross-Similarity Matrices (CSMs) between Blocks

Given a set of M beat-synchronous block features for a song A and a set of N beat-synchronous block features for a song B, we compare all pairs of blocks between the two songs for that feature set, yielding an $M \times N$ cross-similarity matrix (CSM), which can be used to align the songs. For each song, we have 3 different CSMs for the three different feature sets, which are each aligned at the same beat intervals. For MFCC and MFCC SSMs, we use the Euclidean distance (Frobenius norm) to create the CSM, while for HPCP, we use the cosine distance after OTI. We then use the Smith Waterman algorithm [28] to find the best locally aligned sections between a pair of songs. To apply Smith Waterman, we turn the CSM into a binary matrix B^M, so that $B^M_{ij} = 1$ if CSM_{ij} is within the κN^{th} smallest values in row i of the CSM and if CSM_{ij} is within the κM^{th} smallest values in column j of the CSM, and 0 otherwise, where $\kappa \in (0, 1)$ is a *mutual nearest neighbors* threshold. As shown by [25] and [29], this is a crucial step for improving robustness.

Once we have the B^M matrix, we use a diagonally constrained variant of Smith Waterman to locally align the two songs. The score returned by this algorithm roughly corresponds to the length of the longest interval of consecutive blocks matched between songs, with more tolerance for gaps than a naive cross-correlation. More details can be found in[25] and [29]. For comparison, we perform alignments with the CSMs obtained for each feature individually and on the CSM obtained from fusing them.

3 Feature Fusion

Figure 2 shows an overall pipeline for the fusion process. We first briefly review the general mathematical technique that we use to fuse cross-similarity matrices from different feature sets, and then we show specifically how we apply it in our pipeline.

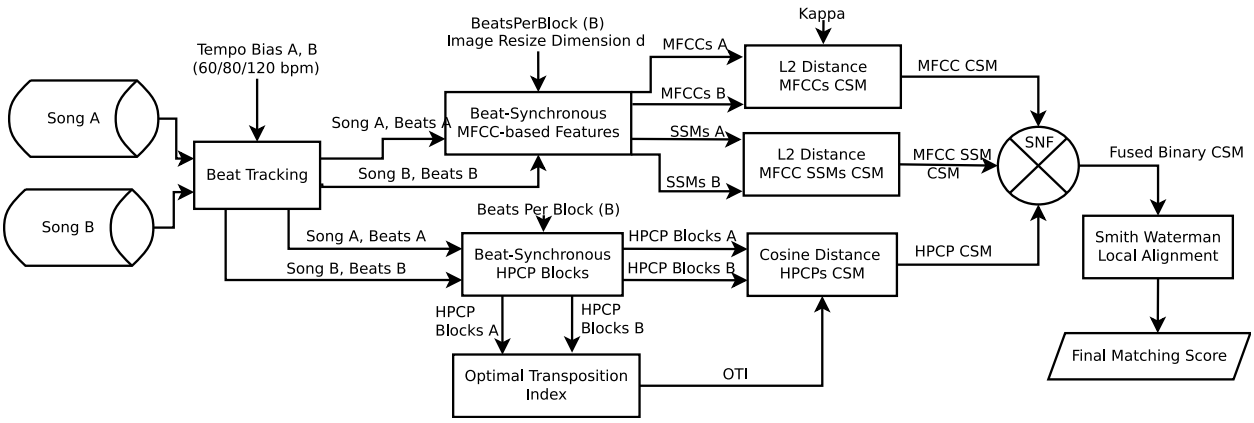

Figure 2: A block diagram of our system which performs early similarity network fusion of blocked MFCCs, MFCC SSMs, and HPCPs before scoring with Smith Waterman alignment.

3.1 Similarity Network Fusion (SNF)

Given all pairs of similarity scores between objects using different features, Similarity Network Fusion (SNF) is designed to find a better matrix of all pairwise scores that combines the strengths of each feature [32, 33]. Roughly, it performs a random walk using the nearest neighbor sets from one feature while using the transition probability matrices from all of the other features, while continuously switching which feature set is used to determine the nearest neighbors. More precisely, the algorithm is defined as follows. First, start with a pairwise distance function $\rho(i, j)$ between objects for each feature type, and create an exponential kernel $W(i, j) = e^{-\rho^2(i,j)/2(\sigma_{ij})^2}$, where σ_{ij} is a neighborhood-autotuned scale which is the average of $\rho(i, j)$ and the mean k-nearest neighbor intervals of blocks i and j, for some k (see [33] for more details). Now, create the following Markov transition matrices

$$P(i,j) = \left\{ \begin{array}{cc} \frac{1}{2}\frac{W(i,j)}{\sum_{k\neq i} W(i,k)} & j \neq i \\ 1/2 & \text{otherwise} \end{array} \right\} \tag{1}$$

This is simply a first order Markov chain with a regularized diagonal to promote self-recurrence. Once this matrix has been obtained, create a truncated k-nearest neighbor version of this matrix

$$S(i,j) = \left\{ \begin{array}{cc} \frac{W(i,j)}{\sum_{k\in N(i)} W(i,k)} & j \in N(i) \\ 0 & \text{otherwise} \end{array} \right\} \tag{2}$$

where $N(i)$ are the k nearest neighbors of vertex i, for some chosen k. Now let P^f and S^f be the P and S matrices for the f^{th} feature set, and let $P^f_{t=0} = P^f$. Then define a first order "cross-diffusion" random walk recursively as follows

$$P^f_{t+1} = S^f \left(\frac{\sum_{v\neq f} P^v_t}{m-1} \right) (S^f)^T \tag{3}$$

where m is the total number of features. In other words, a random walk is occurring but with probabilities that are modulated by similarity kernels from other features. As shown by [32], this process will eventually converge, but we can cut it off early. Whenever it stops, the final fused transition probabilities are $\hat{P}_t = \frac{1}{m}\sum_{k=1}^{M} P^k_t$.

3.2 Late SNF

One way to use SNF is to let the matrix $\rho(i, j)$ be all pairwise distance between songs, computed by some alignment [6]. In this case, the result should be a better network of similarity scores between all songs [3].

[3]Note that [27] essentially do the same thing with only one feature set

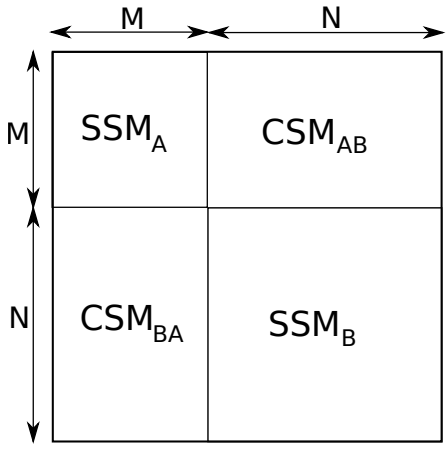

Figure 3: A pictorial representation of the SSM that results when concatenating song B to song A, which we feed to SNF for early fusion of low level features. Including self-similarity blocks of each song helps to promote structural elements in cross-similarity regions during SNF.

We follow a similar approach to [6], but we we work with the Smith Waterman scores we get from a unique combination of MFCC, MFCC SSM, and HPCP blocks ([6] applied SNF to different alignment schemes on the same feature set). Given a particular score matrix S between all pairs of songs, we compute the kernel matrix W as $W(i,j) = 1/S(i,j)$. Since Smith Waterman gives a higher score for better matching songs, this ensures that the kernel is close to 0 in this case. At this point, we perform SNF, and we obtain a final $N \times N$ transition probability matrix P. We can then look along each row to find the neighboring songs with maximum fused probability. This process can be thought of as exploiting the *network* of all songs in a collection in an unsupervised manner.

3.3 Early SNF

In addition to SNF after Smith Waterman has given scores, we can perform fusion at the feature level before running Smith Waterman. One advantage of doing fusion before scores are computed is that we don't need a network of songs to compute a score; we can obtain an improved score between two songs without any other context[4]. Our technique for early fusion, which we found to be superior to the "OR fusion" proposed in [13], is to apply SNF on the cross-similarity matrices obtained from two or more different feature sets before creating a binary CSM and sending it off to Smith Waterman.

As defined in Section 3.1, SNF operates on self-simlarity matrices (SSMs), so it cannot be directly applied to this problem. To make it so that CSMs fit into the framework, we create a "parent SSM" for each feature set that holds both SSMs and the CSM for that feature set. In particular, given song A with M blocks in a particular feature set and song B with N blocks in that feature set, form the SSM D_{AB} which is the SSM that results after concatenating song B to the end of song A. Let the SSM for song A be D_A, the SSM for song B be D_B, and the CSM between them be C_{AB}. Then D_{AB} can be split into four sub-blocks:

$$D_{AB}(i,j) = \left\{ \begin{array}{cc} D_A(i,j) & i < M, j < M \\ D_B(i-M, j-M) & i >= M, j >= M \\ C_{AB}(i, j-M) & i < M, j >= M \\ C_{BA}(i-M, j) = & \\ C_{AB}^T(j, i-M) & i >= M, j < M \end{array} \right\} \quad (4)$$

Figure 3 shows this pictorially. Given such a matrix for each feature set, we could then run SNF and extract the cross-similarity sub-matrix at the end. The issue with this is the dynamic range of the SSM may be quite different from the dynamic range of the CSM, as it is likely that blocks in song A are much more similar to other blocks in song A than they are to blocks in B. To mitigate this, given a nearest

[4]Note that [6] refer to SNF after Smith Waterman as "early fusion" with respect to rank aggregation, which they call "late fusion," but we call their technique "late fusion" because we fuse before Smith Waterman with SNF, which is even earlier in the pipeline.

Table 1: Results of different features and fusion techniques on the Covers 80 dataset.

	MR	MRR	Top-01	Top-10	../80
MFCCs	29.7	0.538	79	97	42/80
SSMs	15.1	0.615	91	111	48/80
HPCPs	18.2	0.673	102	119	53/80
Late SSMs/MFCCs	14.0	0.7	107	125	55/80
Late All	8.63	0.824	127	141	64/80
Early	7.76	0.846	131	143	68/80
Early + Late	7.59	0.873	136	144	69/80
[6]	?	0.625	?	114	?

neighbor threshold κ for the CSM, we compute the kernel neighborhood scales σ_{ij} individually for D_A, D_B, and CSM_{AB}, and we put them together in the final kernel matrix W_{AB} according to Figure 3. Once we have such a matrix W_{AB} for each feature set, we can finally perform SNF. At the end, we will end up with a fused probability matrix P, from which we can extract the cross-probability $P_{C_{AB}}$. We can then take mutual highest probabilities (akin to mutual nearest neighbors) to extract a binary matrix and perform Smith Waterman as normal. Figure 4 shows an example of constructed matrices W_{AB} and the resulting fused probabilities P.

One advantage of this technique is that since the CSM and SSMs are treated together and normalized to a similar range, any recurrent structure which exists in the SSMs can reinforce otherwise weaker structure in the CSMs during the diffusion process. This can potentially help to strengthen weaker beat matches in an otherwise well-matching section, leading to longer uninterrupted diagonals in the resulting binary CSM.

3.4 Early Fusion Examples

Before we launch into a more comprehensive experiment, we show a few examples of early SNF to illustrate the value added. In each example, we used 20-beat blocks, $\kappa = 0.1$ for both similarity fusion and binary nearest neighbors, and 3 iterations of SNF. Figure 5 shows an example where the three individual features are rather poor by themselves, but where they happen to all pick up on similarities in complementary regions. As a result, early SNF does a fantastic job fusing the features. Figure 6 shows an example where MFCC SSMs happen to do better than HPCP, but where the results fusing both are still better than each individually.

4 Experiments

We are now ready to evaluate the performance of this new algorithm. In all of our experiments below, we settle on $\kappa = 0.1$ (the mutual nearest neighbor threshold for binary CSMs) and $B = 20$ beats per block. For similarity network fusion, we take 20 nearest neighbors for both early fusion and late fusion, we perform 3 iterations for early fusion, and we perform 20 iterations for late fusion. We also include an "early + late" fusion result, which is applying late fusion to the network of similarities obtained from all of the feature sets (MFCCs, MFCC SSMs, HPCPs) plus the network of similarities obtained from the early fusion of the three feature sets.

4.1 Covers 80 Dataset

To benchmark our algorithm, we begin by testing it on the "Covers 80" dataset [10]. This dataset contains 160 songs which are split into two disjoint subsets A and B, each with exactly one version of a pair of songs, for a total of 80 pairs. [8] and [11] assess performance as follows: given a song in group A, declare its cover song to be the top ranked song in set B, and record the total number of top ranked songs that are correct. To get a better idea of the performance, we also compute the mean rank (MR), mean reciprocal rank (MRR), and the number of songs correctly identified past a certain number. All of these statistics are computed on the full set of 160 songs, which is more difficult than simply looking in set A or set B.

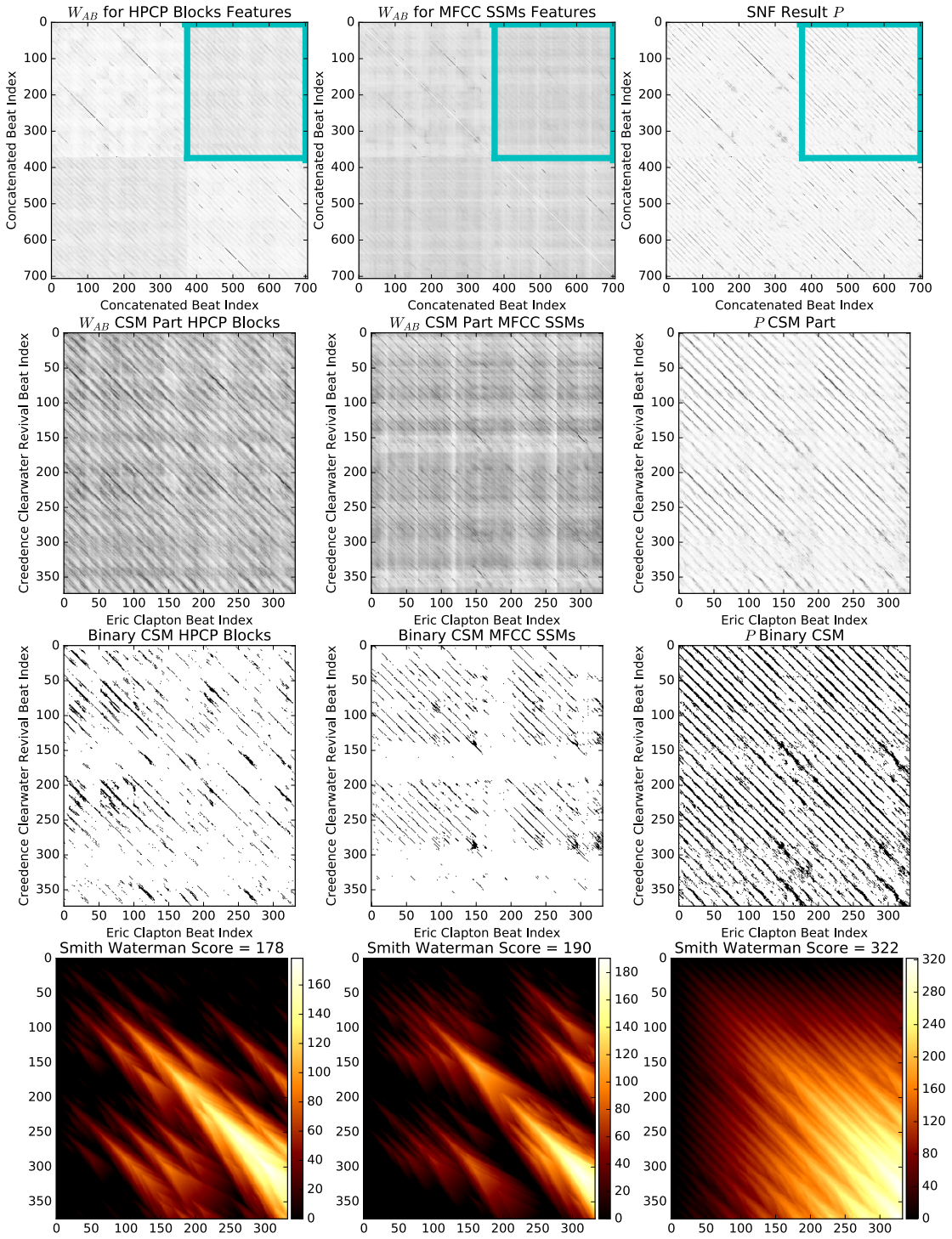

Figure 4: An example of early SNF on blocks of MFCC SSMs and blocks of HPCP features on the song "Before You Accuse Me" with versions by Eric Clapton and Creedence Clearwater Revival. The block size is 20 beats, and there are three iterations of SNF. The kernels W_{AB} are shown for each, and the CSM portion is highlighted with a blue box. The final fused probability matrix P returned from SNF is shown in the upper right. The corresponding CSM portions for all three matrices shown for each on the bottom. In the fused probability matrix, the diagonal regions are much crisper and more distinct from the background than they are for the individual feature sets. The result is that the mutual nearest neighbors binary CSM has longer uninterrupted diagonals, which is reflected by a higher Smith Waterman score.

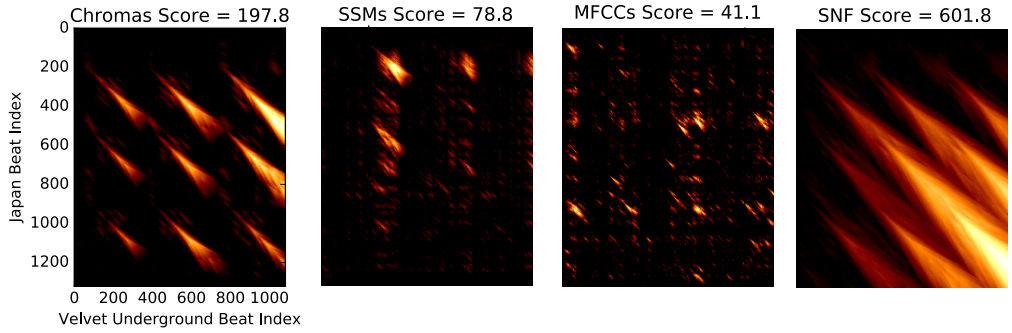

Figure 5: Smith Waterman tables/scores for "All Tomorrow's Parties" by Japan and Velvet Underground.

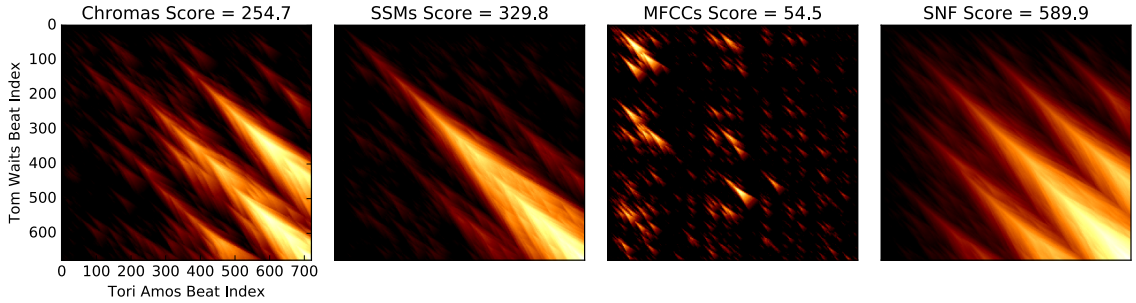

Figure 6: Smith Waterman tables/scores for "Time" by Tom Waits and Tori Amos

Table 1 shows the results. By themselves, HPCP features perform better than MFCC-based features, which is consistent with findings in the literature. However, there are big improvements when fusing them all. Surprisingly, we obtain a score of 42/80 just by blocking and normalizing the MFCCs. This shows the power of having stacked delay MFCCs and of normalizing within each block to cut down on drift. Also, when fusing MFCCs and MFCC SSMs with late fusion, we get a large performance boost over either alone, showing that SSMs are adding complementary information to the MFCCs they summarize.

4.2 Covers 1000 Dataset

To test our algorithm more thoroughly, we created our own dataset by manually choosing 1000 cover songs (395 cliques total) based on annotations given by users on `http://www.secondhandsongs.com`[5]. This dataset covers over a century of Western music from 1905 - 2016, and hence, it covers a wide variety of genres and styles. Figure 7 shows the full distribution of years covered. By contrast, the Covers80 dataset contains almost exclusively pop music from the '80s and early '90s.

Most cliques have only two songs as in the Covers80 dataset, but there are a few cliques with 3 and 4

[5]MFCC and HPCP features for our dataset are publicly available at `http://www.covers1000.net`, along with beat intervals and other metadata including song title, album, and year

Table 2: Results of different features and fusion techniques on the Covers 1000 dataset.

	MR	MRR	Top-01	Top-10
MFCCs	83.3	0.618	583	679
SSMs	72.5	0.623	581	698
HPCPs	44.4	0.757	727	809
Late	19.8	0.875	855	931
Early	22.5	0.829	798	884
Early + Late	14	0.904	884	950

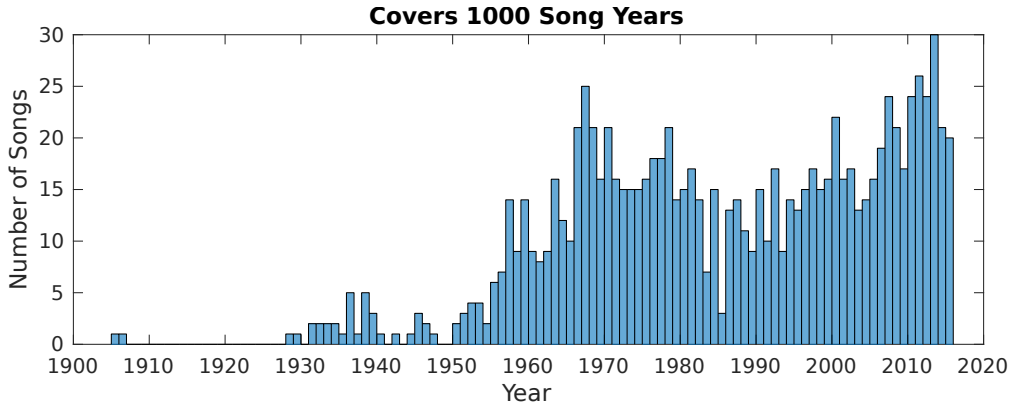

Figure 7: A distribution of years of songs in the Covers 1000 dataset.

songs. In this case, we report the MR and MRR of the first correctly identified song in the clique. Table 2 shows the results. Similar trends are seen to the Covers80 case, and performance scales to this larger size. One difference is that late fusion on HPCPs/MFCCs/MFCC SSMs performed better relative to early fusion, likely because the network was much richer with the additional volume of songs.

4.3 Frank Zappa: "The Black Page"

In our final experiment, we test a clique of 8 cover versions of the song "The Black Page" by Frank Zappa, which is entirely a drum solo that has absolutely no harmonic content. We query each song against all of the songs in the Covers1000 dataset, and we compute the mean average precision (MAP) for the songs in the clique. Unsurprisingly, for HPCP, the MAP is a mere 0.014, while for the rest of the features these songs are quite distinct from the rest of the songs in the Covers 1000 dataset. The best performing feature set is early SNF, with a MAP of 0.98, followed by raw blocked MFCCs at a MAP of 0.97, followed by MFCC SSMs with a MAP of 0.905.

5 Discussion

In this work, we have demonstrated the benefit of combining complementary features at a very early stage of the cover song identification pipeline, in addition to the late fusion techniques in [6]. Unlike [6] and other techniques, our algorithm works on a pair of songs and does not need a network of songs to improve performance, though we show that incorporating information from a network of songs ("late fusion") can further improve results. We showed that HPCP and MFCC features capture complementary information and are able to boost performance substantially over either alone. In the process, we also developed a novel cross-similarity fusion scheme which was validated on several datasets, and which we believe could be useful beyond cover song identification in music structure analysis.

The main drawback of our technique is the requirement of beat tracking. In practice, beat trackers may not return correct onsets. Our current best remedy for this is to use different tempo biases, which blows up computation by a factor of 9. Also, coming up with a single beat level is ill-posed, since most music consists of a hierarchy of rhythmic subdivisions [22]. There does seem to be a recent convergence of techniques for rhythm analysis, though, [7, 18] so hopefully our system will benefit.

In addition to imperfect beat intervals, there are also computational drawbacks in low level alignment, which is why most recent works on cover songs perform approximations to global cross-correlation, such as 2D Fourier Magnitude Coefficients [12, 19]. By contrast, we rely on Smith Waterman, which is a quadratic algorithm, and early SNF adds another quadratic time complexity algorithm even with sparse nearest neighbor sets. To address this, we are in the process of implementing GPU algorithms for every step of our pipeline, and we hope to apply it to the "Second Hand Songs Dataset," which is a subset of the Million Songs Dataset [3].

6 Acknowledgements

Christopher Tralie was partially supported by an NSF Graduate Fellowship NSF under grant DGF-1106401 and an NSF big data grant DKA-1447491. We would also like to thank Erling Wold for pointing out the 8 covers of "The Black Page" by Frank Zappa, and we would like to thank the community at `www.secondhandsongs.com` for meticulously annotating songs which helped us to design Covers 1000.

References

[1] Juan Pablo Bello. Audio-based cover song retrieval using approximate chord sequences: Testing shifts, gaps, swaps and beats. In *ISMIR*, volume 7, pages 239–244, 2007.

[2] Juan Pablo Bello. Grouping recorded music by structural similarity. In *ISMIR 2009, Kobe, Japan, 2009*, pages 531–536, 2009.

[3] Thierry Bertin-Mahieux, Daniel PW Ellis, Brian Whitman, and Paul Lamere. The million song dataset. In *ISMIR*, volume 2, page 10, 2011.

[4] Dmitry Bogdanov, Nicolas Wack, Emilia Gómez, Sankalp Gulati, Perfecto Herrera, Oscar Mayor, Gerard Roma, Justin Salamon, José R. Zapata, and Xavier Serra. Essentia: An audio analysis library for music information retrieval. In *ISMIR 2013, Curitiba, Brazil, 2013*, pages 493–498, 2013.

[5] Ning Chen, J Stephen Downie, Haidong Xiao, Yu Zhu, and Jie Zhu. Modified perceptual linear prediction liftered cepstrum (mplplc) model for pop cover song recognition. In *ISMIR*, pages 598–604, 2015.

[6] Ning Chen, Wei Li, and Haidong Xiao. Fusing similarity functions for cover song identification. *Multimedia Tools and Applications*, pages 1–24, 2017.

[7] Norberto Degara, Enrique Argones Rúa, Antonio Pena, Soledad Torres-Guijarro, Matthew EP Davies, and Mark D Plumbley. Reliability-informed beat tracking of musical signals. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):290–301, 2012.

[8] Daniel PW Ellis. Identifying'cover songs' with beat-synchronous chroma features. *MIREX 2006*, pages 1–4, 2006.

[9] Daniel PW Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36(1):51–60, 2007.

[10] Daniel PW Ellis. The "covers80" cover song data set. *URL: http://labrosa. ee. columbia. edu/projects/coversongs/covers80*, 2007.

[11] Daniel PW Ellis and Courtenay Valentine Cotton. The 2007 labrosa cover song detection system. *MIREX 2007*, 2007.

[12] Daniel PW Ellis and Bertin-Mahieux Thierry. Large-scale cover song recognition using the 2d fourier transform magnitude. In *The 13th international society for music information retrieval conference*, pages 241–246, 2012.

[13] Rémi Foucard, J-L Durrieu, Mathieu Lagrange, and Gaël Richard. Multimodal similarity between musical streams for cover version detection. In *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pages 5514–5517. IEEE, 2010.

[14] Emilia Gómez. Tonal description of polyphonic audio for music content processing. *INFORMS Journal on Computing*, 18(3):294–304, 2006.

[15] Emilia Gómez and Perfecto Herrera. The song remains the same: identifying versions of the same piece using tonal descriptors. In *ISMIR*, pages 180–185, 2006.

[16] Jaap Haitsma and Ton Kalker. A highly robust audio fingerprinting system. In *Ismir*, volume 2002, pages 107–115, 2002.

[17] Holger Kantz and Thomas Schreiber. *Nonlinear time series analysis*, volume 7. Cambridge university press, 2004.

[18] Florian Krebs, Sebastian Böck, and Gerhard Widmer. An efficient state-space model for joint tempo and meter tracking. In *ISMIR*, pages 72–78, 2015.

[19] Oriol Nieto and Juan Pablo Bello. Music segment similarity using 2d-fourier magnitude coefficients. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 664–668. IEEE, 2014.

[20] Julien Osmalsky, Jean-Jacques Embrechts, Peter Foster, and Simon Dixon. Combining features for cover song identification. In *16th International Society for Music Information Retrieval Conference*, 2015.

[21] Julien Osmalsky, Marc Van Droogenbroeck, and Jean-Jacques Embrechts. Enhancing cover song identification with hierarchical rank aggregation. In *Proceedings of the 17th International for Music Information Retrieval Conference*, pages 136–142, 2016.

[22] Elio Quinton, Christopher Harte, and Mark Sandler. Extraction of metrical structure from music recordings. In *Proc. of the 18th Int. Conference on Digital Audio Effects (DAFx). Trondheim*, 2015.

[23] J Serra. Music similarity based on sequences of descriptors: tonal features applied to audio cover song identification. *Department of Information and Communication Technologies, Universitat Pompeu Fabra, Barcelona, Spain*, 2007.

[24] Joan Serra, Emilia Gómez, and Perfecto Herrera. Transposing chroma representations to a common key. In *IEEE CS Conference on The Use of Symbols to Represent Music and Multimedia Objects*, pages 45–48, 2008.

[25] Joan Serra, Emilia Gómez, Perfecto Herrera, and Xavier Serra. Chroma binary similarity and local alignment applied to cover song identification. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(6):1138–1151, 2008.

[26] Joan Serra, Xavier Serra, and Ralph G Andrzejak. Cross recurrence quantification for cover song identification. *New Journal of Physics*, 11(9):093017, 2009.

[27] Joan Serrà, Massimiliano Zanin, Perfecto Herrera, and Xavier Serra. Characterization and exploitation of community structure in cover song networks. *Pattern Recognition Letters*, 33(9):1032–1041, 2012.

[28] Temple F Smith and Michael S Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981.

[29] Christopher J Tralie and Paul Bendich. Cover song identification with timbral shape sequences. In *16th International Society for Music Information Retrieval (ISMIR)*, pages 38–44, 2015.

[30] Avery Wang. The shazam music recognition service. *Communications of the ACM*, 49(8):44–48, 2006.

[31] Avery Wang et al. An industrial strength audio search algorithm. In *ISMIR*, pages 7–13. Washington, DC, 2003.

[32] Bo Wang, Jiayan Jiang, Wei Wang, Zhi-Hua Zhou, and Zhuowen Tu. Unsupervised metric fusion by cross diffusion. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2997–3004. IEEE, 2012.

[33] Bo Wang, Aziz M Mezlini, Feyyaz Demir, Marc Fiume, Zhuowen Tu, Michael Brudno, Benjamin Haibe-Kains, and Anna Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nature methods*, 11(3):333–337, 2014.