A Self-Similarity Approach to Repairing Large Dropouts of Streamed Music

JONATHAN DOHERTY, KEVIN CURRAN, and PAUL MCKEVITT, University of Ulster

Enjoyment of audio has now become about flexibility and personal freedom. Digital audio content can be acquired from many sources and wireless networking allows digital media devices and associated peripherals to be unencumbered by wires. However, despite recent improvements in capacity and quality of service, wireless networks are inherently unreliable communications channels for the streaming of audio, being susceptible to the effects of range, interference, and occlusion. This time-varying reliability of wireless audio transfer introduces data corruption and loss, with unpleasant audible effects that can be profound and prolonged in duration. Traditional communications techniques for error mitigation perform poorly and in a bandwidth inefficient manner in the presence of such large-scale defects in a digital audio stream. A novel solution that can complement existing techniques takes account of the semantics and natural repetition of music. Through the use of self-similarity metadata, missing or damaged audio segments can be seamlessly replaced with similar undamaged segments that have already been successfully received. We propose a technology to generate relevant self-similarity metadata for arbitrary audio material and to utilize this metadata within a wireless audio receiver to provide sophisticated and real-time correction of large-scale errors. The primary objectives are to match the current section of a song being received with previous sections while identifying incomplete sections and determining replacements based on previously received portions of the song. This article outlines our approach to Forward Error Correction (FEC) technology that is used to "repair" a bursty dropout when listening to time-dependent media on a wireless network. Using self-similarity analysis on a music file, we can "automatically" repair the dropout with a similar portion of the music already received thereby minimizing a listener's discomfort.

Categories and Subject Descriptors: E.4 [**Coding and Information Theory**]: Error control codes; Data compaction and compression

General Terms: Performance

Additional Key Words and Phrases: Streaming audio, streaming media, forward error correction, audio repair, data compaction and compression

ACM Reference Format:

Doherty, J., Curran, K., and McKevitt, P. 2013. A self-similarity approach to repairing large dropouts of streamed music. ACM Trans. Multimedia Comput. Commun. Appl. 9, 3, Article 20 (June 2013), 20 pages.
DOI: http://dx.doi.org/10.1145/2487268.2487273

1. INTRODUCTION

Streaming media across the Internet is still an unreliable process. Current technologies for streaming media have reached optimum potential in respect of compression (both lossy and lossless) and buffering songs streamed from a Web-based server to clients. However, the rapid uptake of wireless

Author's addresses: J. Doherty, K. Curran (corresponding author), P. McKevitt, School of Computing and Intelligent Systems, University of Ulster, Derry, Northern Ireland; email: kj.curran@ulster.ac.uk.

DOI: http://dx.doi.org/10.1145/2487268.2487273

communications has led to more recent problems being identified. Traffic on a wireless network can be categorized in the same way as on cabled networks. File transfers cannot tolerate packet loss but can take an indefinite length of time. “Real-time” traffic can accept packet loss, within limitations, but must arrive at its destination in a given time frame. Forward Error Correction (FEC) which usually involves redundancy built into the packets, and Automatic Repeat Request (ARQ) Perkins et al. [1998] are two key techniques currently implemented to overcome the problems encountered. However, bandwidth restrictions limit FEC solutions and the “real-time” constraints limit the effectiveness of ARQ.

The focus of this work is to propose a new method of repair for streaming music over bandwidth-constrained networks. One method overlooked to date, which can work alongside existing audio compression schemes, is to take account of the syntax of the music. Songs in general exhibit standard structures that can be used as a forward error correction mechanism. This work implements a system called SoFI (Song Form Intelligence) that determines packet loss and uses previously received portions of the song to predict what possible match already received exists. In turn, this is used in place of the missing packet(s) before the buffer is empty by applying and improving state-of-the-art theories and techniques in pattern matching with a syntactic, semantic, and cognitive approach.

2. MUSIC INFORMATION RETRIEVAL (MIR)

Research in the field of MIR has gathered momentum over the past decade. With the increase of audio file sharing across heterogeneous networks and streamed audio across the Internet, a need has arisen for more accurate search/retrieval of files. Research in the analysis of audio has led to the development of systems that can browse audio files in much the same way as search engines can browse Web pages retrieving relevant data based on specific qualities [Chai and Vercoe 2003; Gomez et al. 2003; Leman et al. 2002]. MIR covers a large area of research topics that range from computational methods for classification, clustering, and modeling to research into music perception, cognition, and emotions. Two inherent problems associated with MIR are the complexity of audio and the complexity of the query [Downie 2004]. Music is a combination of pitch, tempo, timbre, and rhythm, making analysis of music more difficult than text. Structuring a query for music is made difficult owing to the varying representations and interpretations including natural transitions in music. Monophonic style queries usually perform better where simple note matching can be used whereas polyphonic audio files and queries simply compound the problem.

The vast majority of songs in western nations contain a structure known as a song form. A song form is basically a framework which makes the song listenable [Jackson 2008]. Most songs in western music are structured into a Verse (V) and a Chorus (C). The purpose of a verse is to tell the story or describe the feeling. The chorus is generally the focal point of the song, the central theme. A Bridge (B) is a kind of fresh perspective, a small part that may consist of only music, or both lyrics and music, usually placed after the second chorus and often varying in major/minor chords. These are the main parts that are used when describing song form. Often we can describe songs in terms such as having a VCVC form, or a VCBVC form. Not all songs, however, follow a verse, chorus, and bridge pattern. In the history of music the oldest song form is often referred to as folk, where there is no chorus, or any other part, for that matter. In describing a folk song form, or any song that has only verses, the song form is VVVV.

Adding to the complexity of music structure and query structure is the method of audio analysis. The format of an audio file limits its type of use since different file formats exist to allow for better reproduction, compression, and analysis. Hence it is also true that different digital audio formats lead to different methods of analysis. Musical Instrument Digital Interface (MIDI) files were created to distribute music playable on synthesizers of both the hardware and software variety among artists

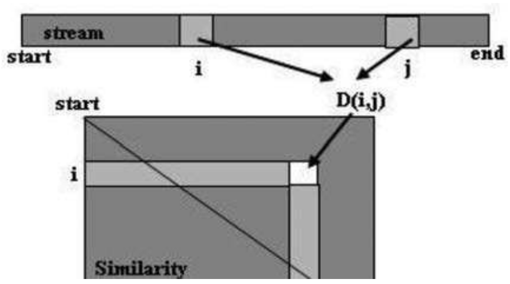


Fig. 1. Embedding an audio stream into a 2D similarity matrix [Foote and Cooper 2003].

and equipment, and because of its notational style the MIDI format allows analysis of pitch, duration, and intensity [Doraisamy and Ruger 2004].

However, reproduction of an MIDI file can vary greatly on different machines simply from differences between the equipment of the composer and the listener. This makes it unsuitable for general audio playback. An alternative to MIDI as a format for analysis is Pulse Code Modulation (PCM), a common method for storing and transmitting uncompressed digital audio. Since it is a generic format, it can be read by most audio applications similar to the way a plain text file can be read by word-processing applications. Owing to the file size analysis can be time consuming when compared to MIDI files. PCM is generally stored on computers in WAV format, which was built into operating systems since the release of Windows 95 thereby making it the de facto standard for sound on PCs. This format for storing sound in files in PCs was developed jointly by Microsoft and IBM. Recent work within the area of similarity identification using polyphonic music has shown that similarity within different sections of a piece of music can aid in both pattern matching for searching large datasets and pattern matching within a single audio file [Foote and Cooper 2003; Meredith et al. 2001; Dannenberg and Hu 2003]. Results from analysis of an audio stream are stored in a similarity matrix created by Foote and Cooper [2003] which can be seen in Figure 1. The similarity matrix is generated by measuring the difference between row and column for the same data. Data along the diagonal will have an exact similarity, but any comparisons "off" the diagonal give a measure of how similar the two values are. Analysis is performed using short time Fourier transform to determine the spectral properties of the segmented audio, this is a variation of the discrete Fourier transform which allows for the influence of time as a factor. Bartech and Wakefield [2002] used chroma-based spectrum analysis technique to identify the chorus or refrain of a song by identifying repeated sections of the audio waveform with the results also being stored in a similarity matrix.

3. AUDIO ANALYSIS

One of the most common formats for audio compression is mp3, defined by the Moving Picture Experts Group (MPEG). MPEG-7 is an international standardized description of various types of multimedia information [Martinez et al. 2002]. Whereas MPEG-4 defines the layout and structure of a file and codecs, MPEG-7 is a more abstract model that incorporates a markup language to define description schemes and descriptors: the Description Definition Language (DDL). Using a hierarchy of classification allows different granularity in the descriptions. While a Google/AltaVista search engine does not exist for audio, many researchers are discovering ways to automatically locate, index, and browse audio using recent advances in technologies such as speech recognition and machine learning. MPEG-7 as a descriptive tool can greatly enhance this area by adding additional metadata based on the content of the audio as well as standard keyword descriptors [Crysandt 2004].

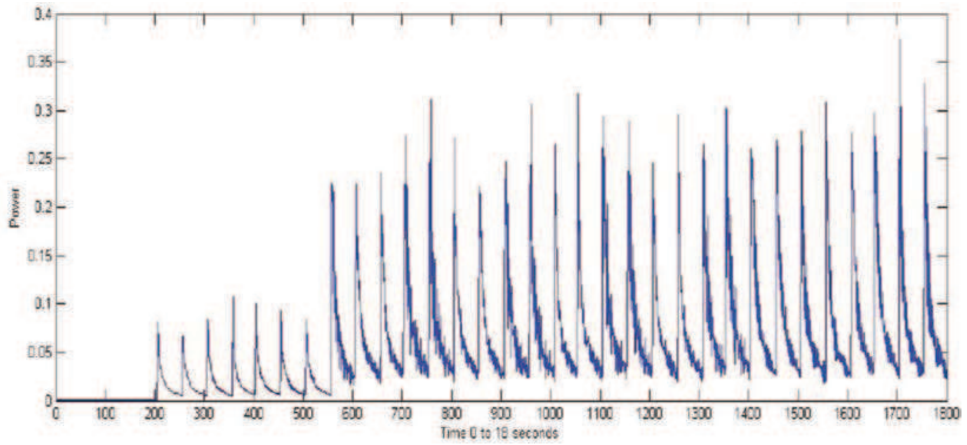


Fig. 2. Example MPEG-7 audio power representation.

3.1 MPEG-7 Descriptors

Through the combination of descriptors, description schemes, and a Description Definition Language (DDL), MPEG-7 can facilitate efficient searching and filtering of files. Descriptors define the syntax and semantics of audio. Descriptors assign specified features to the relevant set of extracted values. Description schemes define the structures of relationships between descriptors, and in a hierarchal manner they can define other description schemes. The Description Definition Language (DDL) defines the syntax to represent audiovisual data results in XML format. Extracted audio features can aid application queries based on more than simple metadata stored within audio files. MPEG-7 descriptors primarily describe low-level features such as color, texture, motion, audio energy, and attributes of the audio/video such as location, time, and quality.

The resulting output is a compact representation of the analyzed audio. Through specific filters on MPEG-7 descriptors a user can specify the type of audio content he/she wants to listen to without any requirement to analyze the audio itself. For example, in one method of similarity analysis [Crysandt 2004] used the mean of the audio power and the mean of the audio spectrum flatness as criteria for a similarity comparison between songs. There are 17 Low-Level Descriptors (LLD) that can appear in a variety of combinations that look at either the temporal or spectral domain of an audio signal. The two basic descriptors are the AudioWaveform (AW) and AudioPower (AP) presented as temporarily sampled scalar values. The AudioWaveform descriptor gives a minimum/maximum value of the signal range within a specified temporal resolution. The AudioPower descriptor provides a measure of the square of the waveform values, thereby providing a simplified representation of the signal showing peaks where the signal has a higher amplitude. A comparative study of some of the LLDs by Lukasiak [2003] shows that when facilitating comparison between audio segments, the AW performs better based on the principle that it has two measurements for each frame as opposed to a single value from the AP analysis. An example giving the first 18 s of a guitar playing a rendition of the 12 Bar Blues is shown in Figure 2. When compared, the level of detail in the AW representation is higher regarding the content of the audio file.

3.1.1 *Audio Spectrum Envelope (ASE).* The Audio Spectrum Envelope (ASE) is a log-frequency power spectrum that can facilitate generation of a reduced spectrum of the original audio. This is performed by summing the energy of the power spectrum within a series of frequency bands. Bands are equally distributed between two frequency edges: loEdge and hiEdge. Default values of 62.5 Hz and 16 KHz correspond to the lower/upper limit of hearing.

3.1.2 *Audio Spectrum Flatness (ASF).* Audio Spectrum Flatness (ASF) describes the flatness properties of the spectrum of an audio signal within a given number of frequency bands. The flatness of a band is defined as the ratio of the geometric mean, that is, the central tendency of a vector, and the arithmetic mean of the spectral power coefficients within the band. The most proficient use of ASF is for musical instrument identification is Essid et al. [2004] who used the ASF as one of the key characteristics for musical instrument recognition. The ASF features, when combined with other feature vectors including the Mel-Frequency Cepstral Coefficients (MFCC), result in high accuracy recognition for instruments belonging to the different families even over short-term decision lengths.

3.1.3 *Audio Spectrum Basis/Projection.* The Audio Spectrum Basis descriptor is a container for basis functions for projecting a spectrum onto a lower-dimensional subspace suitable for probability model classifiers (e.g., neural networks and hidden Markov models). The reduced basis consists of decorrelated features of the spectrum with salient information described more efficiently than with the direct spectrum representation. This reduced representation is suited for probability model classifiers that typically perform best when the input features consist of fewer than 10 dimensions. Audio Spectrum Basis features perform better for sound recognition tasks (e.g., voice, instruments, environmental sounds, animals) than similarity or classification tasks [Casey 2002]. Kim et al. [2004] evaluate the efficiency of audio indexing and retrieval systems based on a combination of the Audio Spectrum Basis (ASB) and Audio Spectrum Projection (ASP) of MPEG-7 audio descriptors.

3.1.4 *Audio Spectrum Centroid (ASC).* The Audio Spectrum Centroid (ASC) is the center of gravity of a log-frequency power spectrum. Unlike the previous MPEG-7 low-level descriptors, the ASC is of a scalar type and provides a high level of dimensional reduction at the cost of high information loss. The ASC provides information on the shape of the power spectrum and indicates whether a power spectrum is dominated by low or high frequencies. The ASC can be regarded as an approximation of the perceptual sharpness of the signal by indicating where the center of mass of the spectrum is. Perceptually, it has a robust connection with the impression of brightness of a sound [Schubert et al. 2004]. Seo et al. [2005] applied the ASC to audio fingerprinting. An audio fingerprint is applied to recognizing audio in the same way a human fingerprint is applied to identify an individual. Fingerprints are perceptual features (short summaries) of a multimedia object, and can be useful in applying search/retrieval queries and copyright detection.

4. SOFI: AUTOMATIC REPAIR OF STREAMED MUSIC OVER BURSTY NETWORKS

This section discusses Song Form Intelligence (SoFI), an intelligent music repair system that repairs dropouts in broadcast audio streams on bursty networks. Unlike other Forward Error Correction (FEC) approaches that attempt to repair errors at packet level, SoFI applies self-similarity in masking large bursty errors in an audio stream received by the listener. SoFI utilizes the MPEG-7 content markup as a base representation of audio and clusters these into similar groups comparing large groupings for similarity. It is this similarity identification process that is used on the client side to replace dropouts in the audio stream being received. The architecture of SoFI is shown in Figure 3 illustrating a client/server approach to audio repair. Figure 3 illustrates the pattern identification components on the server and the music stream repair components on the client. On the server side of Figure 3 is a representation of the feature extraction process prior to the audio being streamed. The feature extractor analyzes the audio from the audio database prior to streaming and creates a results file which

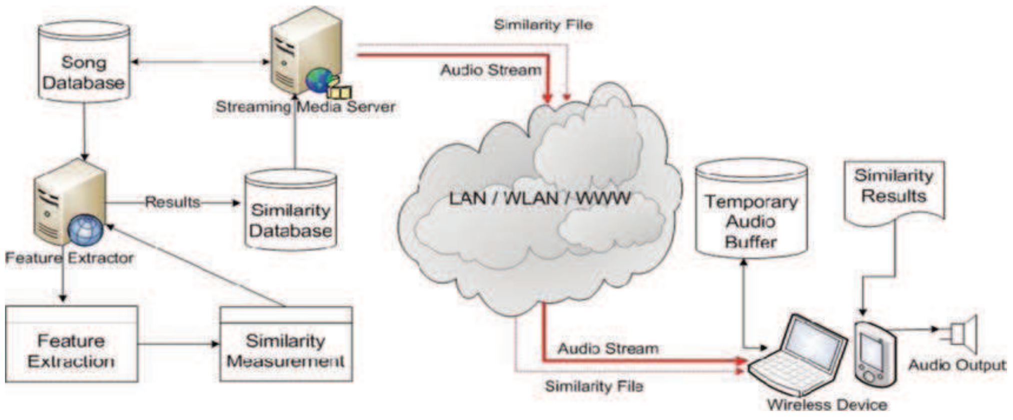


Fig. 3. Architecture of SoFI.

is then stored locally on the server ready for the song to be streamed. The streaming media server then streams the relevant similarity file alongside the audio to the client across the network. On the client side the client receives the broadcast and monitors the network bandwidth for delays of the time-dependent packets.

When the level of the internal buffer of the audio stream becomes critically low the similarity file is accessed to determine the best previously received portion of the song to use as a replacement until the network can recover. The best matching portion of the song is retrieved from a temporary buffer stored on the client machine specifically for this purpose. In a typical MIR system similarity assessment is performed in three stages. First, there is *data reduction*, then *feature extraction,* and finally, *similarity comparisons*. One of the key aspects of feature extraction is to maintain as high a level of reduction as possible without the loss of pertinent data. SoFI makes use of MPEG-7 features in the audio spectrum envelope representation. Songs stored in the database are analyzed and the content description generated from the audio is stored in XML format reducing the file size by up to 90%. This represents a vast reduction in the volume of information to be classified but still retains sufficient information for similarity analysis purposes.

4.1 Audio Spectrum Envelope Feature Extraction

The settings used for extraction stipulate a low and high edge threshold set to 62.5 Hz and 16 KHz, respectively. These settings are the upper and lower bounds of the human auditory system [Pan et al. 1995]. Sounds above and below these levels are of little value and present no additional information that can be utilized when extracting the frequencies. Experiments with values above and below these levels produced results with no gain and more detrimental output as the resultant data was clouded with noise that did not belong to the audio being analyzed. It should be noted that the Joanneum Research facility [MPEG-7 2008] recommend, these settings to be used as default values. Within the low and high frequencies a resolution of 1 is set. This gives a full octave resolution of overlapping frequency bands which are logarithmically spaced from the low-to high-frequency threshold settings. Full octave resolution is the even spread of the frequencies between the low-edge and high-edge limits. The output of the logarithmic frequency range is the weighted sum of the power spectrum in each logarithmic subband. The spectrum, according to a logarithmic frequency scale, consists of one coefficient representing power between 0 Hz and low-edge, a series of coefficients representing power in logarithmically spaced bands between low edge and high edge, and a coefficient representing power above high-edge resulting in 10 samples for each hop of the audio.

As recommended by MPEG-7 [Casey 2001], ASE is used for general, purpose search queries. The Audio Spectrum Projection (ASP) and Audio Spectrum Basis (ASB) are used for more specific search criteria, that is, a specific sound. The ASE features have been derived using a hopsize of 10 ms. and a frame size of 30 ms. This enables overlapping of the audio signal samples to give a more even representation of the audio as it changes from frame to frame. As a rule, using more overlap will provide more analysis points and therefore smoother results over the length of the audio but at the expense of computational costs since more overlap requires more calculations. SoFI generates the ASE descriptions offline for each audio file stored. Audio files are in wav format, thereby ensuring that the audio is of the best possible quality. Lossy compression codecs such as mp3 or ogg can introduce unnecessary noise into the compressed audio that did not exist in the original, even at a high bitrate level. Investigations into the file format to use also showed no improvement in the time to perform the ASE extraction on .mp3/.ogg files compared to wav thereby providing no reasonable gain/justification for their use. ASE was chosen as it maintains almost all of the valuable data in the original polyphonic audio while reducing the dataset to a manageable size. Principal Component Analysis (PCA) Latent Class Analysis (LCA), and Mel-Frequency Cestrums (MFCC) work well when using a training set to recognize a specific voice/sound and audio is "filtered" to specific frequencies. The ASE, however, represents the contents of the audio in a format that lends more easily to similarity identification through clustering. Other MPEG-7 low-level basic spectral descriptors include Audio Spectrum Centoid (ASC), Audio Spectrum Spread (ASS), and Audio Spectrum Flatness.

4.2 Clustering the Audio Spectrum Envelope (ASE)

SoFI uses k-means clustering as a method of identifying similarities within different sections of audio. Using a set number of clusters derived from iterative experimentation of the ASE data provides sufficient grouping. The ASE data files contain a varying number of vectors depending on the length of the audio, but as each vector contains a finite value in that each sample contains a variable quantity that can be resolved into components, an optimal value of $k = 50$ clusters is used. This enables a reasonable computational process with the minimum processing power possible while maintaining maximum variety. Experiments with values above this value produced little or no gain, and with processing time increasing exponentially with each increase in cluster number, were considered too computationally expensive. The k-means output results in an array of numbers of 1..x, where x is the number of samples in the ASE representation ranging from 1 to 50. A file lasting 30 s will result in 3,000 clustered samples, and a file of duration 2 minutes, 45 s will produce 16,500 clustered samples. At this stage of the similarity computation process the cognitive representation of music can be constructed from the output. Where the human mind automatically detects rhythm and repeating patterns, the clustered output notation can be considered similar in that each sample has been compared to all other ASE samples and grouped accordingly. Whereas Jackendoff [1987] presents a hierarchical tree as a representation/notation, a k-means representation conveys a similar representative meaning but on a more detailed linear scale. It is also worth noting that k-means clustering of the ASE is used to obtain a single value for each feature frame as the clustering approach takes into account what precedes and follows the specific sample of audio rather than simply presenting a single value based on its contents.

4.3 Similarity Measurement

Having an audio file classified and clustered into groups is the preliminary step in determining similarity between large sections of the file. Where the ASE is a minimalist data representation/description and the k-means grouping is a cluster representation of similar samples at a granular level, SoFI makes use of a traditional string matching approach to identify large sections of audio. By contrast k-means clustering identifies and groups 10-ms vectors of audio but this needs to be expanded to a larger window in order to facilitate network dropouts. For example, bursty errors on networks can last for as long as 15 to 20 s [Yin et al. 2006; Nafaa et al. 2008] which would mean that if SoFI attempted to apply one identified cluster at a time to repair the gap then it would need to perform the following steps up to 2,000 times.

—Determine the time-point of failure.

—Analyze the current cluster.

—Replace the current section with a suitable previous section.

This is not a feasible option in respect of computational costs. Also, jitter would become a major contributing factor in the resultant audio output to the listener. Applying string matching allows large sections of the k-means cluster output to be compared for overall similarity and the best-effort match can be stored for reference. This file is then used on the client side for reference at a later time on the client machine when dropouts occur. To reduce unnecessary computation, SoFI only compares the clusters in previous sections for similarities. This is based on the principle that when attempting a repair SoFI can only use portions of the audio already received. Any sections beyond this have not yet been received by the client and hence cannot be used. This reduces analysis comparisons considerably in early sections of the audio but as the time-point progresses the number of comparisons increases exponentially. If we find that a section of audio with low streaming quality is not similar to any previous sections of audio, we still use it, that is, if only one section preceding the current section is identified (but with a low similarity rating) then it must be used if a dropout occurs.

The unsupervised k-means clustering method has been shown effective in producing reliable clustering results for many practical applications and is commonly applied to music [Kamata and Furukawa 2007; Peeters et al. 2002; Berenzweig et al. 2004]. The basic steps of k-means clustering are straightforward. First, determine number of clusters k and assume the centroid or center of these clusters. Any random objects can be taken as the initial centroids or the first k objects in sequence can be defined as the initial centroids. The steps are as follows.

(1) Begin with a decision on the value of k = number of clusters.
(2) Put any initial partition that classifies the data into k clusters. These can be assigned randomly, or systematically as the first k data of clusters and assign each of the remaining (N-k) data to the cluster with the nearest centroid.
(3) Take each sample in sequence and compute its distance from the centroid of each of the clusters. If a sample is not currently in the cluster with the closest centroid, switch this sample to that cluster and update the centroid of the cluster gaining the new data and the cluster losing the data.
(4) Repeat step 3 until convergence is achieved, that is, until a pass through the data causes no new assignments.

Since the location of the centroid is initialized to a default value at the start it needs to be adjusted based on the current updated data. All the data is then assigned to the new centroid. This process is repeated until no data is moved to another cluster. This loop can be proved convergent, the convergence will always occur if for each switch in step 2 the sum of distances from each object to that object's group centroid is decreased and there are only finitely many partitions of the objects into k clusters.

One of the advantages of using k-means as a similarity metric is its computational speed in comparison to other approaches. Tao et al. [2004] present a query-by-singing-based musical retrieval system that utilizes k-means clustering. To improve system efficiency, Tao et al. [2004] reorganized their database with a two-stage clustering scheme in both time space and feature space with a k-means algorithm and reported results of over 30% increase in accuracy with a speedup of more than 16 in average query time. Similar to most other algorithms, k-means clustering has a number of weaknesses.

—If the volume of data is small, initial grouping will determine the cluster significantly.

—The number of clusters, k, must be determined beforehand.

—There is never a real cluster, i.e., using the same data, if the data is input in a different order it may produce a different cluster, depending on the volume of data.

—k-means is sensitive to initial conditions. A different initial condition may produce different results.

—It is impossible to know which attribute contributes more to the grouping process since it is assumed that each attribute has the same weight.

—Data with extreme distance measures from the centroid may pull the centroid away from the actual one.

One way to overcome these weaknesses is to use the median instead of mean, also known as k-median clustering. However, this leads to new complications, and can often lead to degradation in performance as shown by Steinbach et al. [2000]. The choice is entirely dependent on the nature and volume of data being used.

4.4 Streaming Server: Icecast2 and Ices2

SoFI uses the Ogg Vorbis [Vorbis 2012] audio file format as an audio compression tool for preparing files for broadcast. As with other compression tools there is no error correction within the stream and packet loss will result in a loss of signal. With a proprietary media player when fragmented packets are dropped a resend request is called using the real-time control protocol. However, SoFI differentiates between fragmented packets and network traffic congestion. As with any media player, SoFI makes use of the resend request for corrupt individual packets, where one or two packets have time to be resent which will not affect the overall audio output. However, when large dropouts of 5, 10, or 15 s occur this will be unrecoverable and the audio output is affected. It is at this point that SoFI uses the previously received portions of a song in an attempt at masking this error from the listener. Ices2 [Ice-2 2012] sends audio data to an Icecast2 server for broadcast to clients. Where Icecast2 [Icecast 2012] handles network connections from clients, Ices2 handles the data to be streamed. Ices2 can either read audio data from disk (Ogg Vorbis files), or sample live audio from a sound card and encode it on-the-fly. Ices2 sends the encoded audio to Icecast2 for streaming. Ices2 and Icecast2 are essential to SoFI in that audio streams can be controlled directly for development and testing purposes, including the ability to start, kill, and restart, when required, during development and testing.

4.5 Client-Side Audio Repair: gStreamer

When repairing dropouts in a live audio stream, priority lies in the system's ability to maintain continuity of output audio. It must balance this alongside with seamless switches between real-time streams being received and buffered portions of the audio, while monitoring network bandwidth levels and responding accordingly. On the client side, a media application needs to be aware of traffic flow to the network buffer because in the event that a dropout occurs a timely swap can be achieved before the internal network buffer fails. It also stores locally all previously received portions of audio: a local buffer is required to fill the missing section of audio until the network recovers. It also plays locally stored audio as well as being able to play network audio. The media player needs to be capable of playing audio stored locally on the client machine.

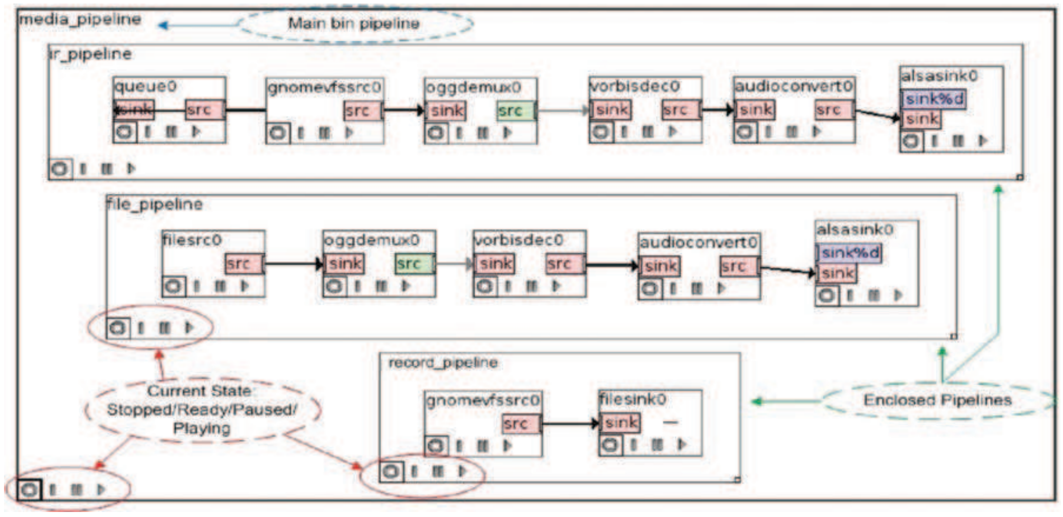


Fig. 4. Graphical representation of SoFI client media handler with multiple pipelines.

Using gStreamer as a media framework, three pipelines have been created to perform these three key requirements simultaneously. Pipelines are a top-level bin which is essentially a container object that can be set to a paused or playing state. The state of any elements contained within the pipeline will assume the stopped/ready/paused/playing pipeline state when the state is set. Once a pipeline is set to playing, it will run in a separate thread until it is manually stopped or the end of the data stream is reached. Figure 4 shows the bin containing the pipelines necessary for the media application to fulfil the three requirements specified earlier. SoFI's media application is programmed in C, has no GUI, and is entirely self-contained. Once it is invoked it requires no interaction from a listener. This is based on the fact that the listener has no control of an audio stream being broadcast, the order in which songs are played, or pausing/rewinding a live stream. Usually an Internet audio stream is shown as simply the length of time that it is connected to the station, not the length of individual songs. SoFI differs in that it resets the GstClock() for each new song. This provides a simple current time-point that enables the media player to determine exactly where in the current song it is and thereby provides a timestamp as a point of reference when network failure occurs.

4.6 Network Monitoring

Built into gStreamer is a message bus that constantly handles internal messages between pipelines and handlers. This message system enables alerts to be raised when unexpected events occur such as end-of-stream and low internal buffer levels. A watch method is created to monitor the internal buffer from the audio stream and when a preset critical level is reached an underrun message is sent to alert the application of imminent network failure. It should be noted that a network failure is not that a network is completely disconnected from the client machine, but is a network connection that is of such poor signal quality (i.e., late packet arrivals leading to jitter) with a low throughput that traffic flow is reduced to an unacceptable level.

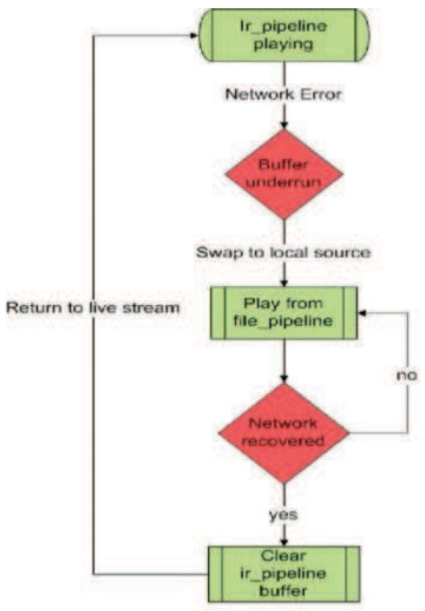


Fig. 5. Flow of control between pipelines.

4.7 Masking Network Dropouts

When the ir_pipeline is playing, the file_pipeline is in a ready state. When a critical buffer level warning is received the media application swaps the audio input from the network to the locally stored file that contains audio from its start point to the point where the network dropout occurred. The threshold is selected based on the characteristics of the network the user is on in addition to the average number of packets received and the rate needed to maintain a "clean" audio signal. When the packet level drops below a critical level the swap will occur. Further development would include monitoring over time and when the rate is consistently slow for a noticeable period of time and the buffer level is not critical but is likely to be soon, then a swap could be performed to allow the buffer to return to a safe level. Figure 5 shows the process of controlling which pipeline is active at any one time. A network failure message calls a procedure that notes the current time-point of the stream and uses this to parse the similarity file already received on the client machine when the current song started. This file contains the output results of the similarity identification previously performed on the server. From this file the previously identified best match section of the audio determines the starting point of the local file on the client machine.

The file_pipeline is now given focus over the ir_pipeline with their states being changed to playing and paused, respectively. After a predetermined length of time the buffer level of the ir_pipeline is checked to determine if network traffic has returned to normal, and if so, then audio output is swapped back to the ir_pipeline. Otherwise file playback continues for the same fixed length of time and is repeated as necessary. In the event that playback of the locally stored file reaches the end of the time-point from when the network failed it is assumed that network traffic levels will not recover and the application ends audio output and closes the pipelines waiting for reinitialization from the user.

4.8 SoFI's Internal Synchronization Clock

The GstClock() in gStreamer is used to maintain synchronization within pipelines during playback. A global clock monitors and synchronizes the pads in each pipeline. The clock time is measured in nanoseconds and always increases. This ensures media playbacks at a specific rate. This rate is not necessarily the same as the system clock rate. For instance, a sound card may playback at 44.1 kHz, but that does not mean that after exactly 1 s according to the system clock, the sound card has played back 44,100 samples. This is only true by approximation. Therefore, pipelines with an audio output use the audiosink as a clock provider. This ensures that 1 s of audio will be played back at the same rate as the sound card plays back 1 s of audio. Whenever a component of the pipeline requires the current clock time, it will be requested from the clock. The pipeline that contains all others contains the global clock that all elements in the pipeline use as a base time. This is the clock time at the point where media playback is starting from zero. SoFI can therefore calculate the stream time and synchronize the internal pipelines accordingly. This provides an accurate measure of the playback time in the currently active pipeline. SoFI's media application, through using its own internal clock, can synchronize swapping between the audio stream and the file stored locally. When a network error occurs the current time-point of the internal clock is used as a reference point when accessing the best match data file.

5. EVALUATION

The purpose of MPEG-7 feature extraction is to obtain a low-complexity description of the content of the audio source. A balanced trade-off between reducing the dimensionality of data and retaining maximum information content must be performed. Too many dimensions cause problems with classification, while dimensionality reduction invariably introduces information loss. Our goal is thus to maximize the amount of information per dimension while maintaining a minimally sized dataset. Clustering procedures are essential tools for unsupervised machine learning. Of the array of clustering methods the k-means clustering is one of the more common choices for solving clustering problems. The choice of starting point of the clusters has a direct impact on the outcome. There is no optimum initial cluster positioning but some work has given consideration to this problem with varying outcomes [Chinrungrueng and Sequin 1995; Bradley and Fayyad 1998] and a common rule of thumb, where the initial cluster centroids are initialized evenly across the data, is the most often proposed solution. Our stance here is not to find some unique, definitive grouping of the ASE output, but rather to obtain a qualitative and quantitative understanding of large amounts of N-dimensional MPEG-7 data by finding similarity within it by clustering the audio data.

Although the k cluster number is initially arbitrarily defined, consistency between clusters improves as the number of groupings increases. In Figure 6 box B shows the k cluster of 50 predominantly classified as the same cluster, whereas the 30 and 40 clusters produced more varied classifications. Tests involving k clusters over 50 produced similar results but created large increases in processing time. As noted previously the number of clusters is set to 50. Values above this offer no gain for the level of detail attained. Figure 6 shows a 5-s sample of audio with clusters of 30, 40, and 50 plotted. The different groupings for each 10-ms sample can be seen in both box A and box B. Depending on the number of k clusters specified each sample will be classified differently. The majority of samples using 30 clusters are shown in green (*), the red samples (+) are for a value of 40 clusters, and the 50-cluster grouping is shown in blue (x). In box A a distinct difference between the values can be seen, where the 30-cluster group has been identified as predominantly between 0 and 5. The 30-cluster grouping in box A also has a high number associated with groups at the high end of clusters between 25 and 30, which shows a high level of inconsistency between samples.

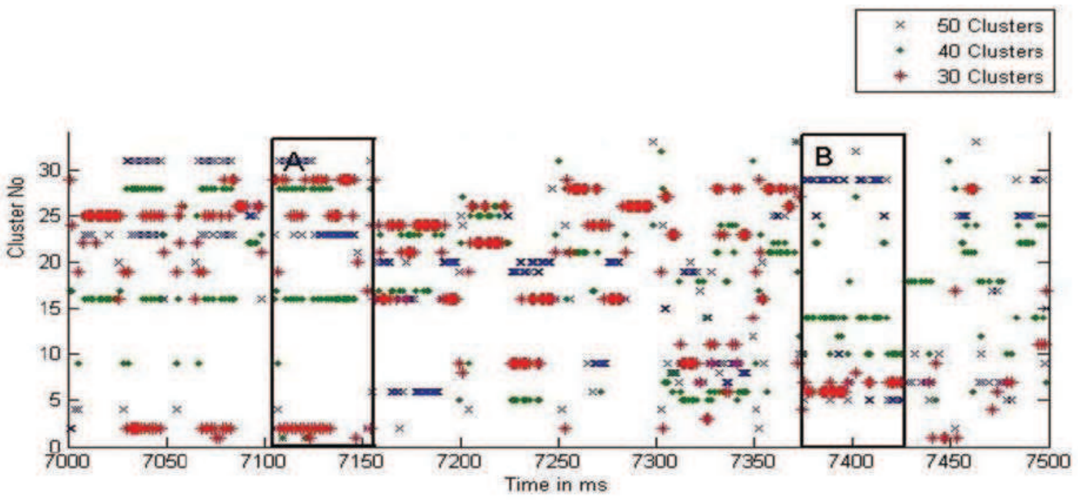


Fig. 6. A comparison of cluster selection.

Table I. k Cluster Computations Relative to Size of k

	Iterations			
k Clusters	Song A	Song B	Song C	Song D
30	8040	3270	4410	6240
40	6520	13280	6650	10160
50	8750	12000	11000	13550
60	10260	17040	15060	16800
80	13520	19680	31360	18080
100	19700	30700	39300	45700

Table I shows the number of calculations required based on the chosen value of k clusters. The number of computations does not increase in a linear or exponential manner, but is based on the complexity of the music and its composition together with the duration of the audio. In this case iterations and computations have the same meaning. The actual number of computations will be dependent on the number of iterations performed when determining the k clusters. Song A requires more calculation due to its composition. Song A is the 12 Bar Blues sample audio used as a testbed throughout this work. Since it contains a high level of variation between time frames, centroids and distances need to be reevaluated more frequently. Songs B, C and D are a random collection from the music collection used within this work. Their basic descriptions are presented in Table II where all songs used for evaluation are listed alphabetically and categorized by genre. The definition of Degree of WTF is merely an indicator of the repetitive nature of the song in relation to the Western Tonal Format (WTF) of popular chart music, that is, how often sections are repeated within the song with no metric of measurement. Most of the songs were released as singles within the United Kingdom, the majority of which reached UK top 40 album chart. Some songs are from albums that reached the UK top 40 album chart and contained at least one "hit single" release.

When a choice of k clusters is set to above 40 a steady increase of computations can be seen in song A (Table I). This is contradicted when using clusters below the level of 40 where songs A,B,C,

Table II. Songs Used in Experiments with Western Tonal Format (WTF) Level

Song	Artist	Genre	Duration (m)	Degree of WTF
12 Bar Blues	N/A	Blues/Jazz	0:37	Medium
All the right friends	R.E.M	Pop/Rock	2:48	High
Anywhere Is	Enya	New Age	3:46	Low
At My Most Beautiful	R.E.M	Pop/Rock	3:46	Medium
Baby one more time	Britney Spears	Pop/Electro	3:31	High
Book of Days	Enya	New Age	2:56	Low
Caribbean Blue	Enya	New Age	3:58	Low
Crazy in Love	Beyonce	Pop/R&B	3:56	High
Daysleeper	R.E.M	Pop/Rock	3:40	High
Don't stop the music	Rihanna	Hip Hop	5:39	Medium
Hole in the Head	Sugababes	Pop	3:38	High
Imitation of Life	R.E.M.	Rock	3:58	Medium
Nine Million Bicycles	Katie Melua	Pop/Jazz/Blues	3:15	High
Orange Crush	R.E.M.	Pop/Rock	3:52	Medium
Orinoco Flow	Enya	New Age	4:26	Low
Stand	R.E.M	Pop/Rock	3:12	High

Table III. Degree of Repetition within Each Song

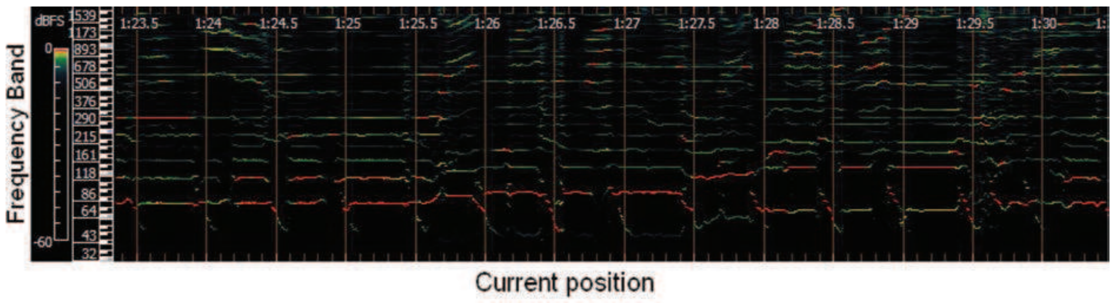
	Audio Properties	
Song	Duration (mins.)	Degree of Western Tonal Format
A	3.86	Medium
B	3.86	Medium
C	3.20	High
D	4.43	Low

and D all produce varying results. This can be partially attributed to the limited number of k clusters that differing values can be assigned to [Zha et al. 2002; Kriegel et al. 2005]. More variations with fewer clusters mean more comparisons since one sample with a high value can offset the centroid of the associated cluster and this needs to be adjusted more frequently. Having a small cluster number means the centroid value for the initially defined cluster will change far more frequently since more "matching" sections of audio will be found and incorporated into the cluster and this will affect the "centroid" for the overall cluster when the centroid value is recalculation based.

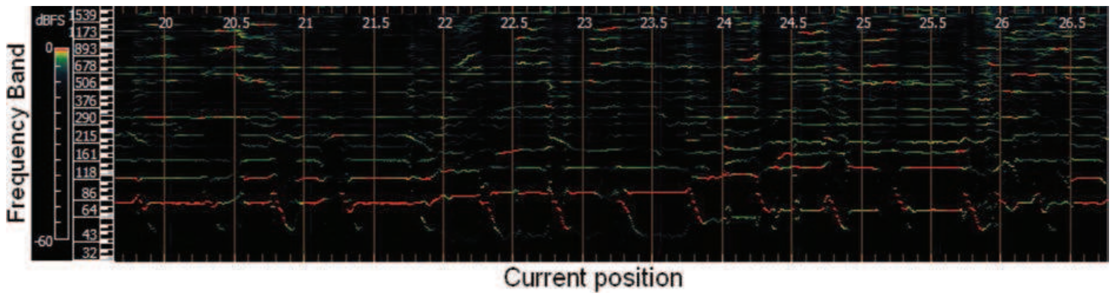
Defining the duration of audio to use as the initial query led to investigations of time lengths between 1 and 5 seconds. Both queries returned the same time-point as the best possible match for the starting time-point of the query. However, additional matches below the best match result using 5 s were found with only 1 s of audio.

This can lead to sections of audio that may be used which are not an accurate replacement for dropouts of over 1 s in length. Using a 5-s length reduces this possibility while increasing the likelihood of the audio following on from the query string time-point still being correct.

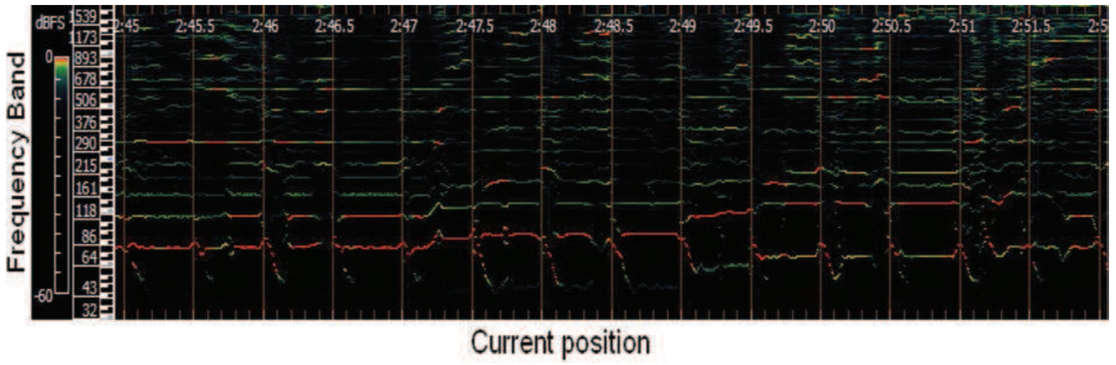
The peak frequency spectrogram presents the precise/dominant frequency of a bin. Within each frequency bin, a single value representing each bin is displayed as a short horizontal line as opposed to the whole block/bin. This allows the strongest details to be seen without interference from neighbouring frequencies. Colors are representative of the power of the particular frequency ranging from black through to red. Figure 7 shows three similar sections identified in a peak frequency representation.



(a) original query section



(b) best match at 20 seconds



(c) best match at 2 minutes 45 seconds

Fig. 7. Peak frequency spectrum representation of three similar audio sections.

From this layout the similarities between the different sections can be seen more clearly. The most evident repetition among the range of frequencies can be seen at the lower end of the scale where strong base tones are more applicable [Olson 1967]. Base drums and male vocals can be more dominant within this range. More evident at this level is the repetition of the frequencies over the fixed temporal length of the sample. Close similarities between duration and power can be seen across graphs (a) to (c) in Figure 7. Clear signs of similarities can be seen visually when comparing the lower frequencies of

Table IV. Test Songs Using Different Approaches to Dropout Repair

	Repair type	Details of repair	Duration of repair (s.)
Song 1	Linear interpolation	N/A	5 s.
2	Down Sampling	8 bit mono @4000 Hz.	5 s.
3	Jitter	0.5 s. continuous sections repeated or moved	5 s.
4	5 s. chorus repair	Earlier chorus section with avg match 0.720	5 s.
5	5 s. best match	Best match ratio of all sections of song 0.409	5 s.
6	Control audio with no repair	N/A	N/A
7	10 s. repair	Repair a 10 s. dropout using a 5 s. match	10 s.
8	Low match ratio 5 s. repair	Use low match ratio of 0.890 as replacement	5 s.

Figure 7(a) with Figure 7(b) and Figure 7(c). The lower frequencies of the audio (shown primarily in red) consist of drum and bass guitar and indicate a clear similarity of strength and duration between the original query section and the closest matches found, thereby indicating repetitive sections of the music. Quantitative evaluations discussed earlier demonstrate how accurate replacements can be when comparing errors. However, not every dropout occurrence can be accurately repaired. Experiments were done to determine whether an acceptable level of repair is performed based on multiple scenarios. Sixteen subjects were invited to participate in this experiment. Over 50% of the test subjects were in the age group of 18 to 25 with over 60% being male. The subjects were asked to rate their musical knowledge on a scale one to ten in order to gain an understanding of their auditory perception skills. A keen musician would be able to discern any irregular changes in a piece of music better than a novice listener. Almost 70% considered themselves as average by scoring their abilities between 4, 5, and 6. Only two subjects (13%) considered themselves as above average with a listening skill of 7 and 9. None of the subjects considered themselves at a skill level for writing music. Using a ten-point Likert scale, test subjects were given eight audio files to listen to in succession and asked to rate the repair in a questionnaire form. Since the tests were to be the same for each listener the audio repair in each song was simulated and stored locally as a wave file. Using sample time-points that have been identified as similar within the collection of songs, specific experiments were executed at predetermined time-points. The point of failure for each song was decided by the particular test, that is, a verse failure at the beginning of the song and even the duration of the dropout. The audio files were then reconstructed based on the identified segments using a wave editor to ensure precise matching of time-points. This removed any possible ambiguity introduced by rerunning the audio broadcasts multiple times with varying network characteristics. Each song was evaluated when it had completed and before the next began. Once all songs had been played at least once a general rank for all the songs was requested from each test subject. This rank was based on an order of preference of one to eight with one being the most favored method of repair and eight the least favored.

Table IV shows the eight different songs used in the evaluation tests. All eight songs follow the Western Tonal Format (WTF) of intro/verse/- chorus/verse/chorus with an additional bridge included. The first song includes linear interpolation to bridge the missing 5-s segment. This was performed by determining the overall frequency of the audio signal at time-point x and time-point y and creating a signal to bridge the gap based on these frequencies. Song 2 includes the low-bandwidth redundancy approach where a second signal is also broadcast but at a much lower sampling rate. When a dropout occurs this lower quality audio has a higher chance of still arriving at its destination. A downsampled segment of the audio was used to replace the dropout at a sample rate that equates to just below telephone audio quality of 8 bit mono at 4000 Hz. Song 3 includes the reordered packet effect that reduces the dropout affect to appear as jitter. However, as this evaluation is performed “offline” a simulation

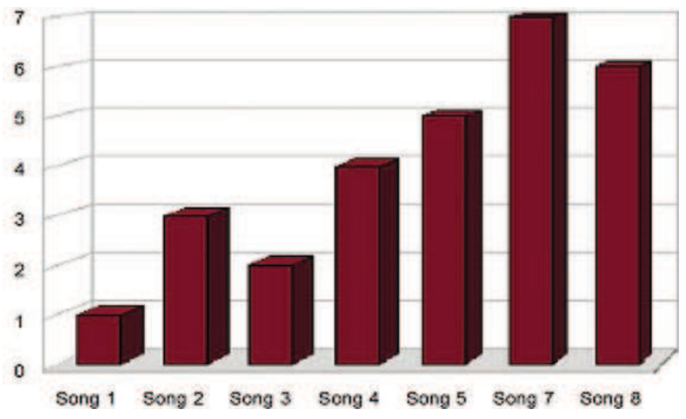


Fig. 8. User evaluation rank for each song.

was created by reordering and repeating small segments of the audio for the duration specified. Songs 4, 5, 7, and 8 were all based on SoFI's best-effort analysis and use varying levels of similar matches as identified during analysis. Although the audio was not broadcast as a live audio stream the effects of the audio replacement are the same. Performing a simulation provides the opportunity to enable subjects to evaluate the best, and worst-case scenarios that SoFI would be expected to repair. Song 6 acted as a control test to evaluate listeners, attention to the audio to ensure random marks were not supplied as a quick response to the survey. The final part of the evaluation questionnaire requested the subjects to list the order of preferred repair based on how successful it was at masking a dropout from the listener. Prior to completing this, subjects were briefed on each repair approach and allowed to listen to the specific section where the error occurred in each song over again. With more attentive perception listeners were more aware of what type of repair they were listening to and ranked each song accordingly based on a scale of 1 to 8. All subjects correctly identified the fact that song 6 had no audio changes and automatically ranked it as top with a score of 8. Figure 8 shows the final mean rank for each method of repair from all subjects. This correlates closely to the mean scores subjects provided. Figure 8 shows ranks of between 4 to 7 assigned to songs 4, 5, 7, and 8 while songs 1, 2, and 3 were ranked the lowest.

The average score for each subject is shown in Figure 9. The blue column represents the average mark across all repair methods. The orange column represents the average given for songs 1, 2, and 3 which use varying types of repair to mask a bursty network dropout. The final column for each listener represents the average score for the four different simulated attempts made by SoFI to replace varying time-points and durations of dropouts in the signal. We found that songs 1, 2, and 3 obtain a lower than average score and that the average score across all listeners is directly related to the scale of marking on a per-listener level. We also found that the average score for each listener for songs 4, 5, 7, and 8 including simulated SoFI repairs are scored significantly higher than songs 1, 2, and 3.

These results show a consensus of opinion across all subjects that, although some level of change in the audio stream is perceived, listeners indicate an increase in level of acceptance in simulated attempts at audio repair by SoFI when presented with current alternatives. The mean evaluation score for other methods of repair was 2.67 out of a possible 10. This is almost half the average for all songs where a mean of 4.55 is shown. The mean score obtained for the attempted best-effort repair under differing circumstances obtained was 7.09. This indicates a much higher level of acceptance

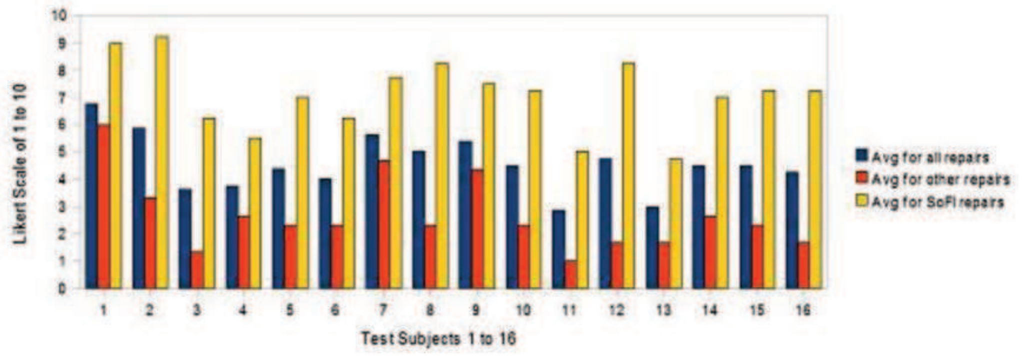


Fig. 9. Comparison of average scores for song repair types: Other approaches, SoFI repairs, and overall average shown.

overall and an almost 200% increase in level of acceptance by the subjects. Looking at the demographic data collected from questions the evaluation questionnaire, no correlation could be found between age/gender with mean scores or the corresponding variance between different scores[1].

Subjective evaluation with 16 test subjects listening to and quantifying simulated dropout scenarios allowed a comparison of SoFI's simulated performance compared to other simulated approaches together with a comparison of how successful SoFI's simulated performance was using different dropout scenarios. The subjective evaluation results showed how a greater level of acceptance of packet dropouts is perceived by subjects with an increased acceptance score of over 200% compared to the other methods.

6. CONCLUSION

Time-dependent streaming media such as Internet radio when combined with wireless networks have a high level of packet loss causing large bursty errors in the music stream. Recent approaches to repairing dropouts in streamed audio include linear interpolation, randomizing the packet order, or encoding additional audio at a higher compression level with the aim to improve throughput. All of these approaches aim to reduce the level of distraction of the dropout to the attention of the listener. However none utilize the natural repetition within a songs structure to replace the missing segment. This article demonstrated the combination of the MPEG-7 audio spectrum envelope feature extraction, k-means clustering, and string matching as a form of self-similarity analysis with the aim to

[1]We are aware of the possibility of a confounding variable in our user evaluation. However, the evaluation of the repairs presented is a subjective listening format, where acceptance of the repair is based on a psychological one as opposed to quantitative evaluation of exactly how the replacement "fits". Based on a subjective evaluation approach the need to use exactly the same song and segment for each repair would almost invalidate the assessment as any subject exposed to the same song will after the fourth or fifth repetition become more of an expert in the composition of the audio and judge repairs more and more harshly. The purpose of the evaluation is to determine an overall level of acceptance without prior knowledge of the audio, along with the principle idea that it is to merely "mask" the dropout that occurred. This could only be achieved through the use of different songs. Alongside this, we believe that the constant repetition of the same song would lead to subjects becoming "disinterested" in the evaluation tests through boredom. An alternative evaluation to subjects listening to a full song would have been to use a single 30-s sample with the 5-s repair included in the middle. However, this would not give sufficient time for a subject to gain an overall "feeling" for the song and would be less able to judge a repair based on what should be the overall knowledge of the song that the subject has gained and developed expectations of what should come next. It is also worth noting that each song used was specifically chosen as each falls neatly in the classification of Western Tonal Format and follows the verse/chorus/verse/chorus pattern thereby making the evaluation of each type of repair as fair and balanced as possible.

repair large dropouts in the time-dependent music stream by using portions of music already received that have been identified as the most similar to the music currently being played and using this as a replacement until the media stream returns.

REFERENCES

Bartsch, M. A. and Wakefield, G. H. 2001. To catch a chorus: Using chroma-based representations for audio thumbnailing. In *Proceedings of the IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*. 15–18.

Berenzweig, A., Logan, B., Ellis, D., and Whitman, B. 2004. A large-scale evaluation of acoustic and subjective music-similarity measures. *Comput. Music J. 28*, 2, 63-76.

Bradley, P. and Fayyad, U. 1998. Refining initial points for k-means clustering. In *Proceedings of the 15th International Conference on Machine Learning*. Vol. 727. Morgan Kaufmann, 91–99.

Casey, M. 2002. General sound classification and similarity in mpeg-7. *Organised Sound 6*, 2, 153–164.

Chai, W. and Vercoe, B. 2003. Structural analysis of musical signals for indexing and thumbnailing. In *Proceedings of the Joint Conference on Digital Libraries*. 27–34.

Chiariglione, L. 2010. Description of mpeg-7 audio low level descriptors. http://mpeg.chiariglione.org/technologies/mpeg-7/mp07-aud(ll)/index.htm

Chinrungruen, C. and Sequin, C. 1995. Optimal adaptive k-means algorithm with dynamic adjustment of learning rate. *IEEE Trans. Neural Netw. 6*, 1, 157–169.

Crysandt, H. 2004. Music identification with mpeg-7. *Proc. SPIE 5307*, 117–124.

Dannenberg, R. B. and Hu, N. 2003. Pattern discovery techniques for music audio. *J. New Music Res. 32*, 2, 153–163.

Downie, J. S. 2004. The scientific evaluation of music information retrieval systems: Foundations and future. *Comput. Music J. 28*, 2, 12–23.

Doraisamy, S. and Ruger, S. 2004. A polyphonic music retrieval system using n-grams. In *Proceedings of the International Conference on Music Information Retrieval*. 204–209.

Essid, S., Richard, G., and David, B. 2004. Efficient musical instrument recognition on solo performance music using basic features. In *Proceedings of the 25th International Conference of the Audio Engineering Society.*

Foote, J. T. and Cooper, M. L. 2003. Media segmentation using self-similarity decomposition. *Proc. SPIE 5021*, 167–175.

Gomez, E., Klapuri, A., and Meudic, B. 2003. Melody description and extraction in the context of music content processing. *J. New Music Res. 32*, 1, 23–40.

Ices2. 2008. Ices2. http://www.icecast.org/ices.php.

Icecast. 2008. Icecast. http://www.icecast.org.

Jackendoff, R. 1987. *Consciousness and the Computational Mind*. MIT Press Cambridge, MA.

Jackson, I. 2008. Song forms and terms: A quick study. http://www.irenejackson.com/form.html.

Kamata, M. and Furukawa, K. 2007. Three types of viewers' favorite music videos. In *Proceedings of the International Conference on Advances in Computer Entertainment Technology*. 196–199.

Kim, H, G., Moreau, N., and Sikora, T. 2004. Audio classification based on mpeg-7 spectral basis representations. *IEEE Trans. Circ. Syst. Video Technol. 14*, 5, 716–725.

Kriegel, H. P., Kunath, P., Pfeifie, M., and Renz, M. 2005. Distributed high-dimensional data. In *Proceedings of the 9th Pacific-Asia Conference on Advances in Knowledge Discovery and Data Mining (PAKDD'05).* Springer.

Leman, M., Clarisse, L., De Baets, B., De Meyer, H., Lesaffre, M., Martens, G., Martens, J. P., and Van Steelant, D. 2002. Tendencies, perspectives, and opportunities of musical audio-mining. *Forum Acusticum Sevilla 33*, 3–4, 1–6.

Lukasiak, J., Stirling, D., Harders, N., and Perrow, S. 2003. Performance of mpeg-7 low level audio descriptors with compressed data. In *Proceedings of the International Conference on Multimedia and Expo (ICME'03).* Vol. 3. 273–276.

Martinez, J. M., Koenen, R., and Pereira, F. 2002. MPEG-7: The generic multimedia content description standard, part 1. *IEEE Multimedia 9*, 2, 78–87.

Meredith, D., Wiggins, G. A., and Lemstrom, K. 2001. Pattern induction and matching in polyphonic music and other multidimensional datasets. In *Proceedings of the 5th World Multiconference on Systemics, Cybernetics and Informatics (SCI'01).* 22–25.

MPEG-7. 2008. MPEG 7 library: A complete api to manipulate mpeg 7 documents. Joanneum Research. http://iiss039.joanneum.at/cms/index.php?id=84.

Nafaa, A., Taleb, T., and Murphy, L. 2008. Forward error correction strategies for media streaming over wireless networks. *IEEE Comm. Mag. 46*, 1, 72–79.

OLSON, H. F. 1967. *Music, Physics and Engineering*. Dover Publications.

PAN, D., INC, M., AND SCHAUMBURG, I. L. 1995. A tutorial on mpeg/audio compression. *IEEE Multimedia 2*, 2, 60–74.

PEETERS, G., LA BURTHE, A., AND RODET, X. 2002. Toward automatic music audio summary generation from signal analysis. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR'02)*. 98–106

PERKINS, C., HODSON, O., AND HARDMAN, V. 1998. A survey of packet loss recovery techniques for streaming audio. *IEEE Netw. 12*, 5, 40–48.

SCHUBERT, E., WOLFE, J., AND TARNOPOLSKY, A. 2004. Spectral centroid and timbre in complex, multiple instrumental textures. In *Proceedings of the 8th International Conference on Music Perception and Cognition*. Society for Music Perception and Cognition.

SEO, J. S., JIN, M., LEE, S., JANG, D., LEE, S., AND YOO, C. D. 2005. Audio fingerprinting based on normalized spectral subband centroids. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*. Vol. 3. 213–216.

STEINBACH, M., KARYPIS, G., AND KUMAR, V. 2000. A comparison of document clustering techniques. In *Proceedings of the KDD Workshop on Text Mining*. 34–42.

TAO, D., LIU, H., AND TANG, X. 2004. K-box: A query-by-singing based music retrieval system. In *Proceedings of the 12th Annual ACM International Conference on Multimedia*. ACM Press, New York, 464–467.

VORBIS. 2008. Ogg vorbis. http://www.vorbis.com.

YIN, J., WANG, X., AND AGRAWAL, D. P. Impact of bursty error rates on the performance of wireless local area network (wlan). *Ad Hoc Netw. 4*, 5, 651–668.

ZHA, H., HE, X., DING, C., GU, M., AND SIMON, H. 2002. Spectral relaxation for k-means clustering. *Adv. Neural Inf. Process. Syst. 2*, 1057–1064.

Received April 2012; revised November 2012; accepted April 2013