MOSAIKBOX: IMPROVING FULLY AUTOMATIC DJ MIXING THROUGH RULE-BASED STEM MODIFICATION AND PRECISE BEAT-GRID ESTIMATION

Robert Sowula
TU Wien
robert@sowula.at

Peter Knees
Faculty of Informatics, TU Wien
peter.knees@tuwien.ac.at

ABSTRACT

We present a novel system for automatic music mixing combining diverse music information retrieval (MIR) techniques and sources for song selection and transitioning. Specifically, we explore how music source separation and stem analysis can contribute to the task of music similarity calculation by modifying incompatible stems using a rule-based approach and investigate how audio-based similarity measures can be supplemented by lyrics as contextual information to capture more aspects of music. Additionally, we propose a novel approach for tempo detection, outperforming state-of-the-art techniques in low error-tolerance windows. We evaluate our approaches using a listening experiment and compare them to a state-of-the-art model as a baseline. The results show that our approach to automatic song selection and automated music mixing significantly outperforms the baseline and that our rule-based stem removal approach significantly enhances the perceived quality of a mix. No improvement can be observed for the inclusion of contextual information, i.e., mood information derived from lyrics, into the music similarity measure.

1. INTRODUCTION

DJs have become an essential aspect of many large social events today. The quality of their performance heavily depends on the DJ's experience, knowledge of music, and understanding of what resonates with the audience [1]. Although many attempts [2–8] have been made to automate this role, a DJ is still considered indispensable for providing enjoyable and seamless listening experiences and mixing, i.e., transitioning of tracks.

In this paper, we propose *Mosaikbox*, an automatic music mixing system primarily focused on EDM, incorporating state-of-the-art MIR methods to mimic aspects considered by DJs when selecting and mixing tracks. For selection, these aspects include timbre, which has been shown to improve methods for judging music similarity if combined with other auditory descriptors [9]. For mixing, a typical transition technique is the fade in/fade out. During its transition period, the two songs are audible for some time. Even if the tempo and key match perfectly and the timbral compatibility is high, a dissimilar drum pattern, e.g., with off-beats at different times than the original track or clashing vocals, can result in a combination that does not sound right. Removing incompatible stems during transitions would solve many traditional mixing challenges.

A rather open question is the use of contextual information for track selection by DJs. Contextual information, such as song lyrics, contains information that audio-based approaches cannot capture and vice versa. Since approaches such as [10] have shown that lyrics can be used to predict the mood of a song, combining audio-based and contextual information might further improve the quality of track selection.

The objectives of this paper are therefore: *(1)* to introduce a novel automatic music-mixing pipeline, *(2)* to investigate how a rule-based stem modification procedure can support a music similarity measure in automatic music mixing, and *(3)* to explore whether we can improve the used musical similarity measure by complementing it with contextual information.

2. RELATED WORK

Besides commercial, closed-source tools, such as VirtualDJ, djay, and NI Traktor, various academic approaches have been proposed for automatic music mixing. Jehan [2] introduced an automated DJ system focused on beat matching on downbeats and transitioning on rhythmically similar segments without incorporating harmonic or timbral information or automatic track selection. Building on this, Lin et al. [3] incorporated pitch information and introduced a method for automatic track selection and ordering. Ishizaki et al. [4] further proposed a method for reducing discomfort when mixing songs with heavily differing tempi in his automatic DJ system. Davies et al. developed AutoMashUpper (AMU) [5], an automatic mashup system that mixes songs using a mashability estimate over phrase-level segments. AMU incorporates a weighted combination of rhythmic and harmonic similarity and spectral balance into its mashability measure. Hiari et al. [6, 7] introduced another automated DJ system based on latent topic modeling of the chroma features and beat similarity for

Attribution: R. Sowula and P. Knees, "Mosaikbox: Improving Fully Automatic DJ Mixing Through Rule-based Stem Modification and Precise Beat-Grid Estimation", in *Proc. of the 25th Int. Society for Music Information Retrieval Conf.,* San Francisco, United States, 2024.

song selection and cue point estimation. Vande Veire and De Bie [11] built an automatic mixing system similar to AMU with multiple transition methods and a focus on musical style similarity, but less powerful similarity measures compared to AMU to optimize for runtime performance. Huang et al. [8] proposed a pure mashup system that uses isolated stems of different songs to create a mashup. Unlike the previously described automated mixing systems, this approach focuses on mixing a combination of stems, ensuring that each stem type is used only once.

Our work differs from existing methods by proposing a more comprehensive mixability measure to capture additional audio and contextual aspects to better match DJ music selection techniques. Furthermore, we focus on working with completely mastered tracks, integrate state-of-the-art MIR techniques, and perform stem separation to support our mixability measure.

3. METHOD

Our method for automatic mixing comprises the following components to build the *Mosaikbox* system: beat grid estimation and tempo detection, structural segmentation, multi-faceted estimation of music similarity, and mixing of tracks.

3.1 Beat Grid Estimation and Tempo Detection

We build our beat and tempo detection pipeline upon a fixed beat grid approach using a 4/4 time signature, similar to popular DJ software. To build a beat grid, we need two types of information: the song's tempo and the location of the first downbeat. We derive the beat positions, including the beat types (1$^{\text{st}}$, 2$^{\text{nd}}$, 3$^{\text{rd}}$, and 4$^{\text{th}}$ beat) using the state-of-the-art beat tracking system *BeatNet* [12].

Calculating the tempo by averaging inter-beat intervals or using their median can lead to octave errors ($\frac{1}{2}$, $\frac{1}{3}$, 2, 3 multiples of the tempo), where the problematic tempi are the $\frac{1}{3}$ and 3 multiples of the true tempo for non-duple meter music. To address this, we model the beat grid estimation as a 2-dimensional constrained minimization problem, given the detected beat timings t_i, where $i = 1, 2, \ldots, n$. Note that some beats might be missing due to detection errors. We want to find the optimal first downbeat position g_1 and the tempo bpm such that the beat positions of the constructed beat grid g_j are evenly spaced and have minimal deviation from the detected beat positions t_i.

To restrict the search space, we estimate the tempo bpm_{est} by using the inter-beat median Δt_{Mdn}. We then perform a global search twice using the dual annealing algorithm, a variant of the simulated annealing algorithm, paired with a local search algorithm for accepted solutions [13]. The objective is to minimize the mean of the absolute differences between each estimated beat grid position g_i and detected beat positions t_j. The initial global search spans a wide range, from 60 bpm to $+15\%$ of bpm_{est} and the first downbeat from 0 to $+40\%$ of Δt_{Mdn}. To avoid local minima, we conduct a subsequent narrower search within $\pm 5\%$ of bpm_{est} and 0 to $+5\%$ of Δt_{Mdn}. Finally, we fine-tune the beat grid by performing local minimization over only the offset of the first downbeat position from 0 to $+40\%$ of Δt_{Mdn}.

3.1.1 Benchmark

We evaluated the performance of our beat grid and tempo estimation algorithm on the GiantSteps dataset [14, 15], as it has not been used for training BeatNet [12] nor current state-of-the-art tempo estimation approaches such as the one by Böck and Davis [16].

Slight deviations in the estimated tempo lead to significant errors in the beat grid estimation. Thus, we deem the metrics *Accuracy 1* and *Accuracy 2* as defined by Gouyon et al. [17] using a 4% tolerance window as too loose, and additionally evaluate the performance of our tempo estimation algorithm for smaller tolerance windows of 1% and 0%. Table 1 compares our tempo estimation algorithm and its inter-beat interval (IBI) pre-estimation with the state-of-the-art tempo estimation algorithm by Böck and Davis [16] on the GiantSteps dataset. While our approach does not outperform the state-of-the-art algorithm for the 4% tolerance window, it demonstrates better performance for the 1% and 0% tolerance windows. The results also show that while IBI is important, it is not the primary contributor to our method's performance.

	Böck & Davis [16]	IBI	Ours
Accuracy 1 (4%)	**87.29**	74.13	82.30
Accuracy 1 (1%)	67.02	58.40	**69.59**
Accuracy 1 (0%)	0.15	3.03	**19.97**
Accuracy 2 (4%)	**96.97**	78.08	90.77
Accuracy 2 (1%)	74.38	61.35	**76.70**
Accuracy 2 (0%)	0.45	3.11	**24.51**

Table 1. Comparison of our tempo estimation algorithm and its inter-beat interval estimation with a state-of-the-art approach on unseen data from the GiantSteps dataset.

3.2 Structural Segmentation

Music transitions sound most pleasing when performed at musically fitting positions of a song. We therefore combined the boundary detection algorithm by Serrà et al. [18] with the labeling approach by Nieto and Bello [19].

In electronic music, segments typically align with downbeats. Therefore, we quantize the detected segment boundaries to the nearest beat position and shift them by one beat to the nearest downbeat. Boundaries starting or ending on the third beat are not shifted, due to potential causes, such as errors in the downbeat detection, time signature estimation, or different song structures.

Although mixing intros with outros is a straightforward way of transitioning whole songs, we abstain from this practice as we aim for a more energetic mix. Thus, we penalize intro and outro segments by the factor 0.5, which is then multiplied by the similarity measure. The progression of the energy level is a task addressed in the similarity measure. We assume that low-energy and high-energy segments will not be mixed and thus do not differentiate between other segment types.

3.3 Music Similarity

3.3.1 Rhythmic Similarity

We believe that drums are the primary rhythmic component in EDM music. Instead of relying on onset detection functions, which have poor performance in polyphonic audio, we employ the drum transcription system by Southall et al. [20, 21] to extract drum patterns from the audio. To be able to detect different kinds of rhythm patterns besides the classical "straight" pattern, such as "swing", "shuffle" or "offbeats" which are a primary component in EDM subgenres such as drum and bass, we follow the AMU approach of Davies et al. [5] and sub-divide the beat grid into 12 equally spaced intervals. We then detect the *kick*, *snare*, and *hi-hat* drum positions, quantize them over the sub-beat grid, and stack them on top of each other to obtain a 3-dimensional binary vector R_n for all songs n of length $K * 12$, where K is the number of beat positions of a song. The rhythmic similarity is then calculated between phrase sections p of the seed song s and a candidate song c for all k beat shifts of c. While AMU uses cosine similarity as a rhythmic similarity measure, we decided to employ a stricter similarity measure to capture dissimilarities in the drum patterns. Thus, we defined the similarity measure as the average of the sub-beat positions where the drum patterns of the seed song section and the candidate song section match.

For each drum vector $R_{s,p,d}$ within phrase section p of the seed song s, where $d \in 1, 2, 3$ denotes the drum vector dimensions corresponding to the *kick*, *snare*, and *hi-hat*, we compute the average number of matching sub-beat positions l over all beat shifts k against all candidate songs c. The overall rhythmic similarity $M_{R,s}(k)$ is then derived by averaging the similarities obtained across the three drum dimensions d as

$$M_{R,c}(k) = \frac{1}{3}\sum_{d=1}^{3}\left(\frac{1}{m}\sum_{l=1}^{m}[R_{s,p,d,l} = R_{c,k,d,l}]\right), \quad (1)$$

where m is the length of the drum vector $R_{s,p}$ of the phrase section in the seed song, and $[R_{s,p,d,l} = R_{c,k,d,l}]$ denotes the Iverson bracket.

3.3.2 Timbral Similarity

To model the timbral component, we will follow the approach of Rocha et al. [22] and Panteli et al. [23], using MFCCs and the auditory descriptors spectral flatness and dirtiness. By stacking the MFCCs, spectral flatness, and dirtiness descriptors on top of each other, we obtain a 28-dimensional vector $T_{n,p}$ for a song n and phrase section p. Due to high computational demands, we calculate the timbral component once per phrase section instead of every beat shift k of the candidate song c, assuming the timbral component remains relatively constant across phrase sections. The timbral similarity is then calculated by computing the cosine similarity between the timbral component $T_{s,p}$ of phrase section p of seed song s and the timbral component $T_{c,q}$ of all phrase sections q of candidate song c as

$$M_{T,c}(q) = \frac{T_{s,p} \cdot T_{c,q}}{\|T_{s,p}\|\|T_{c,q}\|}. \quad (2)$$

3.3.3 Key Similarity

Harmonic compatibility is essential when mixing songs, as it avoids dissonance and supports continuity between songs by enabling smooth transitions. We decided to use the key detection algorithm *KeyFinder* [24] due to its open-source availability and still good performance compared to recent state-of-the-art key detection algorithms. Additionally, we incorporate pitch shifting in the song selection process to be more flexible and less constrained by the harmonic aspect of the songs. As pitch-shifting algorithms can hurt the audio quality [25], we nonetheless want to keep pitch-shifts as small as possible. To this end, we identify key distances.

We define a harmonic key distance measure $D_{K_1}(K_2)$ as the minimum semitone distance between the tonic notes of two keys K_1 and K_2. The key similarity measure $M_{K,c}$ is then defined as

$$M_{K,c} = \begin{cases} 1, & \text{if } D_{K_s}(K_c) = 0 \\ D_{K_s}(K_c)^{-1}, & \text{otherwise} \end{cases}, \quad (3)$$

where $D_{K_s}(K_c)$ is the key distance between the key K_s of the seed song s and the key K_c of the candidate song c.

3.3.4 Harmonic Similarity and Spectral Balance

Harmonic content and the energy across the low-, mid-, and high-frequency bands change throughout a song and thus must be reflected in the similarity measure. We compute the harmonic similarity and spectral balance measure, $M_{H,c}(k)$ and $M_{L,c}(k)$, respectively, based on the approach by Davies et al. [5].

3.3.5 Contextual Similarity

Mixing songs at positions with similar lyrics is a transition technique that could make the transition more related and seamless, independently of audio-based similarity. This method is commonly executed by playing a repeated phrase of the first song and then mixing in the second song with a similar vocal phrase.

Although lyrics are content information, they are often analyzed using contextual methods and are thus treated accordingly [26]. Due to the significant variation in lyrics across song sections, we find classical textual similarity measures such as *TF-IDF* unsuitable for our task. Instead, we capture the lyrics' similarity by extracting the whole lyrics' semantic meaning. We use Reimers and Gurevych [27] approach to compute sentence embeddings C_n over the lyrics of all songs n. The similarity measure $M_{C,c}$ is then calculated by computing the cosine-similarity between the sentence embedding C_s of the seed song s and the sentence embedding C_c of the candidate song c as

$$M_{C,c} = \frac{C_s \cdot C_c}{\|C_s\|\|C_c\|}. \quad (4)$$

3.3.6 *Mixability*

We compute the beat-wise mixability for a candidate song c against the phrase section p of the seed song s by combining the weighted similarity measures of rhythm, timbre, key, harmony, and spectral balance, as follows:

$$M_c(k) = \omega_R M_{R,c}(k) + \omega_T M_{T,c}(q) + \omega_K M_{K,c} + \omega_H M_{H,c}(k) + \omega_L M_{L,c}(k), \tag{5}$$

where q is the phrase section of c corresponding to the beat shift k. The mixability measure considers the 64 beats after the phrase section p of the seed song s instead of the entire phrase section p. This forward-moving approach enables us to maintain a song's dynamics by focusing on the upcoming segments instead of past segments. Through extensive, informal testing, we found the following weights to give the most convincing results: $\omega_R = 0.3$, $\omega_T = 0.75$, $\omega_K = 0.2$, $\omega_H = 0.2$, and $\omega_L = 0.1$.

To incorporate the contextual similarity measure, we extend the audio-based mixability measure $M_c(k)$ by the contextual similarity measure $M_{C,c}$ with the weight $\omega_C = 0.25$, as follows:

$$M'_c(k) = M_c(k) + \omega_C M_{C,c}. \tag{6}$$

Our initial experiments showed that choosing the transition point by selecting the beat shift k with the highest mixability score did not yield satisfactory results. Songs were transitioned at non-downbeat positions or unnatural downbeat intervals (e.g., 7, 9, 15, 17 downbeats), leading to a misaligned mix. To counteract this, we consider only beat shifts k that correspond to the segment boundary q of the candidate songs c and calculate the transition (cue) point as follows:

$$k_{\text{cue}}(c) = \arg\max_{k \in q} M_c(k). \tag{7}$$

We also record the timbral and rhythmic similarity at the transition point, $t_{\text{cue}}(c)$, $r_{\text{cue}}(c)$, and will use this information to improve the equalization in the mixing process.

We compute the song schedule by selecting the candidate song c with the highest mixability and extract the phrase section p for c up to the next segment boundary q, but at least for a minimum of λ_{minPlay}. We found that a λ_{minPlay} value of 55 seconds leads to a good balance between how long a song is played and how often songs are changed. We then select the phrase section p of c as the seed phrase section and repeat the process until the desired length of the mix is reached.

3.4 Mixing

Before transitioning, we first bring the loudness of each song to a consistent level of -14 LUFS. We then pitch-shift the audio to a harmonically compatible key and beat-match the song by time-stretching the audio to the same tempo as the previous song, using a maximal tempo change of $\pm 8\%$ as a limit. We use a transition length of 16 downbeats, where the transition starts with eight downbeats before the song excerpt's end and ends with eight downbeats after the transition point of the current song.

To prevent clashing frequency bands in the mix, we mainly base our equalization process on the ”bass-swap“ technique [28, Chapter 16] and extend it to the high-frequency band as well. A visualization of our standard equalization process is depicted in Figure 1.

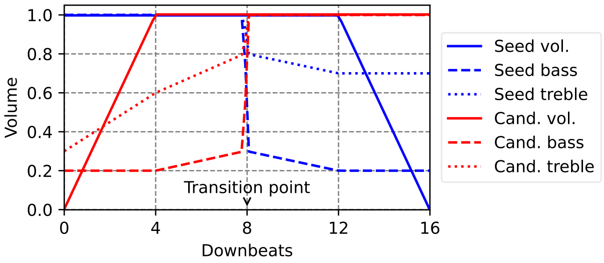


Figure 1. Standard equalization applied to both excerpts.

Overequalization can lead to a dull mix, which is why we will use information from our mixability calculation to identify and adjust problematic frequency ranges. We consider mid frequencies of songs with a dissimilar timbre ($t_{\text{cue}}(c) < 0.95$) as clashing and reduce the mid frequencies of the song that is currently playing, shifting the focus on the mid frequencies to the new song. In case of a high rhythmic similarity ($r_{\text{cue}}(c) \geq 0.95$), we apply less attenuation to the bass frequencies. Finally, we also assume that songs with an attenuated drum stem need even less equalization in the high frequencies, as drums, especially hi-hats, are a primary contributor to the high frequencies. We therefore introduce the high frequencies of the song that are to be mixed in earlier and with less attenuation.

3.4.1 *Rule-based Stem Modification*

We employ the pre-trained music source separation (MSS) model *HT Demucs* [29, 30] to separate the audio into the four stems: vocals, drums, bass, and other.

As previously noted, our tempo estimation algorithm predicts the tempo for only around 25% of songs with perfect accuracy. Even though the rhythmic similarity measure implicitly captures errors in tempo detection, rhythmic compatibility is only one of the components of the mixability measure, thus opening up the possibility of mixing in a rhythmic incompatible song. To counteract this, without entirely excluding rhythmic incompatible songs, we introduce a drum stem modification procedure for songs with rhythmic compatibility below $r_{\text{cue}}(c) < 0.95$.

Further, we generally want to prevent mixing song excerpts containing vocals, as vocal clashing can similarly lead to a reduced mix quality. We detect vocal segments by splitting the vocal stem, obtained by our MSS stage, into boundaries on ”silent“ sections that persist for one second or longer with a loudness below -40 dBFS and filter our vocal segments with a length below 400ms. We consider two song excerpts as clashing if the vocals during the transition intersect for more than two seconds and attenuate the vocals of the currently playing song. Figure 2 depicts the drum stem and vocal modification procedure.

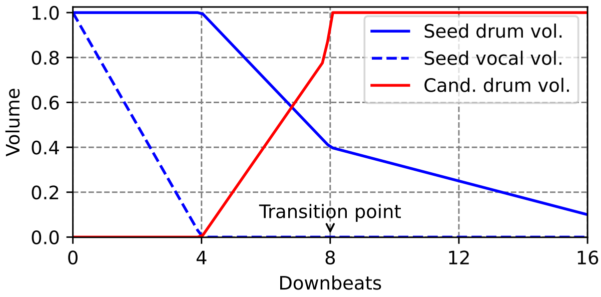


Figure 2. Equalization over the vocal and drum stems.

4. LISTENING EXPERIMENT

Due to the subjective nature of mixes [8] and the lack of ground truth corpora for similarity ratings between songs [22], we will evaluate the performance of our proposed solutions with qualitative methods, specifically using a listening experiment. For this, we developed a web-based survey that facilitates the evaluation of the models by human participants.

4.1 Models

We select AMU by Davies et al. [5] as our baseline model because it aligns with our methodology in prioritizing optimal mixing over runtime compromises and continues to be recognized as a relevant benchmark in recent research, such as [8]. To ensure a fair comparison of our models and counteract the negative influence of mismatched beats, we replace outdated components of AMU with state-of-the-art approaches. In particular, we replace their beat tracking and percussion detection method with our approaches and utilize our mixing procedure to create the mix.

To compare the performance of our models, we evaluate three models: MB_{base}, our approach without the stem modification and contextual information; MB_{stem}, our approach with the stem modification but without contextual information; and MB_{full}, our approach with the stem modification and contextual information (using M'_c). The code of our implementations is available open-source [1] .

Note that, in order to maintain full control over the approaches and integration into a common interface, no commercial tools are included in the evaluation.

4.2 Setup

To understand the impact of musical knowledge on evaluation, we first ask the participants about their musical background and DJing experience. We split the following survey for each model into two parts. In the first part, we gather Song-Pair Compatibility (SPC) ratings by asking the participants to rate the song-scheduling aspect of the models. This allows us to compare the song selection of the models to the collected SPC ratings later on. For all pairs of songs, the participants assess the compatibility based on four categories by answering the following questions: *Timbre*: Are the songs similar regarding timbre? *Rhythm*: Do the songs have a similar rhythmic pattern? *Harmony*: Do the songs have a similar harmonic structure? *Overall Mixable*: Are the songs overall mixable?

In the second part, the participants are presented with the generated mix of a model and are asked to rate the overall quality of each transition of the mix on a scale of 1 (awful) to 5 (excellent), where 3 represents a neutral rating. The models are presented in random order to prevent presentation bias, with no details about the model type disclosed to the participants.

4.3 Dataset

Due to the tempo "lock-in", only songs with a tempo tolerance of maximum $\pm 8\%$ are considered. This commonly results in a genre "lock-in" as well, as songs of the same genre usually have a similar tempo. A preliminary poll among potential participants revealed that most are familiar with the drum and bass genre (DnB). We therefore decided to base our dataset on this genre to make the evaluation more relevant and valid.

We collected a dataset of 250 songs from the most popular DnB playlists of streaming services and randomly sampled 16 songs from this collection to use as input for the mix generation of the models. Out of these 16 songs, we sampled one song as the starting song for all models. To highlight the song selection aspect of the models, we used a top-k approach with k=8 for song selection instead of forcefully mixing all 16 songs.

5. RESULTS AND DISCUSSION

We recruited 30 participants (22 male/8 female), primarily academics aged 23-30 with backgrounds in STEM and economics, 8 of whom had prior experience in DJing. Among the participants, 10 classified their musical background as novice, 13 as intermediate, 7 as advanced, and none stated being a professional musician.

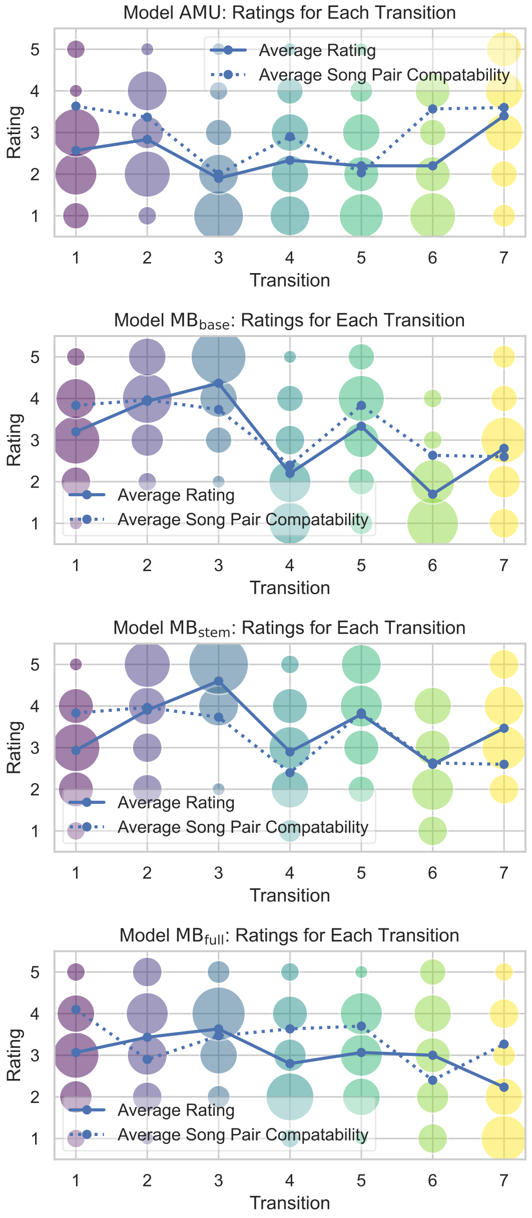
Model	**Transition**	$\textbf{SPC}_{\textbf{Timb}}$	$\textbf{SPC}_{\textbf{Rhy}}$	$\textbf{SPC}_{\textbf{Har}}$	$\textbf{SPC}_{\textbf{Mix}}$
AMU	2.490	0.457	0.505	0.429	0.624
MB_{base}	3.076	0.486	**0.648**	0.505	0.648
MB_{stem}	**3.457**	0.486	**0.648**	0.505	0.648
MB_{full}	3.033	**0.500**	0.619	**0.529**	**0.705**

Table 2. Average transition and SPC ratings for all models from all transitions. MB_{base} and MB_{stem} share identical SPC ratings due to the same song selection.

Table 2 shows that all our models significantly outperformed the AMU baseline in average transition and SPC ratings. The MB_{full} model received the highest SPC ratings for timbre, harmony, and mixability, while the $\text{MB}_{\text{base, stem}}$ models scored higher in rhythm and the MB_{stem} model achieved the best average transition rating.

In Figure 3, we can observe that AMU mostly received negative ratings, while those of MB_{full} had a more consistent distribution, declining towards the end of the mix. Except for the first two transitions, MBstem consistently

[1] https://github.com/robaerd/mosaikbox


Model AMU: Ratings for Each Transition

Model MB$_{\text{base}}$: Ratings for Each Transition

Model MB$_{\text{stem}}$: Ratings for Each Transition

Model MB$_{\text{full}}$: Ratings for Each Transition

Figure 3. Transition ratings across all models, with bubble size indicating the number of ratings per transition and lines representing each transition's average rating and SPC.

outperformed MB$_{\text{base}}$, suggesting that stem separation positively impacts mix quality.

After confirming non-normality with the Shapiro-Wilk test, we test for significant differences using the Friedman test, resulting in a p-value < 0.00001. To determine the best-performing model, we then conduct Wilcoxon signed-rank tests for pairwise comparisons. To account for the family-wise error rate, we correct the p-values using the Holm-Bonferroni method and reject the null hypothesis if the corrected p-value $\hat{p}$ is less than the significance level $\alpha = 0.05$. The results in Table 3 show that all our models significantly outperform AMU, while MB$_{\text{stem}}$ significantly outperforms its base counterpart MB$_{\text{base}}$. No significant difference is found between the MB$_{\text{full}}$ and MB$_{\text{stem}}$ models, which suggests that contextual information does not have a significant impact on the mix quality.

Model 1 (F)	**Model 2 (G)**	$\hat{p}$**-value**$_{F(u)<G(u)}$
MB$_{\text{base}}$	AMU	< 0.0001
MB$_{\text{stem}}$	AMU	< 0.0001
MB$_{\text{full}}$	AMU	< 0.0001
MB$_{\text{stem}}$	MB$_{\text{base}}$	< 0.0001
MB$_{\text{full}}$	MB$_{\text{base}}$	×
MB$_{\text{full}}$	MB$_{\text{stem}}$	×

Table 3. Pairwise tests for significance between models showing corrected p-value levels of the Wilcoxon signed-rank test ('×' means no significance at 0.05 level).

Further significance tests using Mann-Whitney U tests revealed a significant difference in ratings between DJ experience and all musical knowledge levels only for the baseline model AMU, with p-values of 0.001 and 0.0001, respectively. Participants with DJing experience rated the AMU model significantly worse. Analogous, based on the mean ranks of the transition ratings, the higher the musical knowledge level, the worse the rating.

Finally, we tested for significance of the Pearson correlation between transition ratings and the averaged SPC values, indicating a significant strong correlation for MB$_{\text{base}}$ with ($r = 0.83$, $p = 0.02$), suggesting mixes align closely with participant expectations. In contrast, there was a moderate non-significant correlation ($r = 0.6$, $p = 0.1$) for AMU and MB$_{\text{stem}}$ and no significant correlation for MB$_{\text{full}}$ ($r = -0.03$, $p = 0.95$).

The performance gains of our models over AMU in SPC ratings may stem from our rhythmic similarity calculation and the integration of timbral and key similarities into the mixability estimate. Improved transition ratings could be linked to our updated structural segmentation approach. Higher timbre, harmony, and mixability SPC ratings, alongside lower rhythm ratings, might be influenced by mood-related contextual similarities. Lower transition ratings could stem from the lesser relevance of lyrics' semantic meaning in DnB. The listening experiment results are available online. [2]

6. CONCLUSION

In this paper, we proposed the automatic mixing system *Mosaikbox* and demonstrated that it outperforms comparable state-of-the-art systems. We showed that our rule-based stem modification significantly improves the overall mix quality. However, we could not show that including contextual information has any significant positive impact on the mix quality.

Future work will include the impact of new features, such as the energy level of songs and the use of similarity measures obtained by collaborative filtering approaches. In addition, a dynamic transition length will be explored to enhance creativity and adaptability across various genres.

[2] https://github.com/robaerd/mosaikbox-survey

7. ACKNOWLEDGMENTS

This research was funded in whole or in part by the Austrian Science Fund (FWF) (10.55776/P33526). For open access purposes, the author has applied a CC BY public copyright license to any author-accepted manuscript version arising from this submission.

8. ETHICS STATEMENT

One of the main factors in the similarity measure of *Mosaikbox* is timbre, which could lead to a bias towards songs of the same artist or label, neglecting songs of other artists, due to production effects captured by MFCCs (a phenomenon often referred to as album or artist effect [31, 32]). This constitutes a technical algorithmic bias, cf. [33,34]. In the interest of transparency and ethical rigor, we also acknowledge a potential bias in the participant demographics. The participants were mainly academics aged 23-30, with a majority having backgrounds in STEM and economics and a significant portion having familiarity with the drum and bass genre.

Additionally, the rule-based approach of our system offers the advantage of not requiring training on a large dataset of copyrighted music, including DJ interpretations. However, this does not remove the issues related to automating a craft traditionally performed by humans. This raises several concerns, including potential impacts on artistic expression and reception, the role of human creativity, and the future of DJing as a skilled profession.

Furthermore, as with every form of automation, a general adoption of automatic music mixing systems could significantly reduce the demand for DJs, especially in smaller venues. However, automatic music mixing systems, such as ours, can also be used as a tool by DJs to explore new ideas, get suggestions for transitions they might not have thought of, break out of their comfort zone, and increase diversity and creativity in mixes, if designed accordingly.

9. REFERENCES

[1] F. Broughton and B. Brewster, *How to DJ Right: The Art and Science of Playing Records*. Grove/Atlantic, Inc., Dec. 2007.

[2] T. Jehan, “Creating Music by Listening,” Ph.D. dissertation, Massachusetts Institute of Technology, Jan. 2005.

[3] H.-Y. Lin, Y.-T. Lin, and M.-C. Tien, “Music Paste: Concatenating Music Clips based on Chroma and Rhythm Features,” in *Proceedings of the 10th International Society for Music Information Retrieval Conference*. ISMIR, Jan. 2009, pp. 213–218.

[4] H. Ishizaki, K. Hoashi, and Y. Takishima, “Full-Automatic DJ Mixing System with Optimal Tempo Adjustment based on Measurement Function of User Discomfort,” in *Proceedings of the 10th International Society for Music Information Retrieval Conference*. ISMIR, Jan. 2009, pp. 135–140.

[5] M. Davies, P. Hamel, K. Yoshii, and M. Goto, “AutoMashUpper: Automatic Creation of Multi-Song Music Mashups,” *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 12, pp. 1726–1737, Dec. 2014.

[6] T. Hirai, H. Doi, and S. Morishima, “MusicMixer: computer-aided DJ system based on an automatic song mixing,” in *Proceedings of the 12th International Conference on Advances in Computer Entertainment Technology*. Association for Computing Machinery, Nov. 2015, pp. 1–5.

[7] ——, “Musicmixer: Automatic dj system considering beat and latent topic similarity,” in *MultiMedia Modeling - 22nd International Conference*, Q. Tian, R. Hong, X. Liu, N. Sebe, B. Huet, and G.-J. Qi, Eds. Springer International Publishing, 2016, pp. 698–709.

[8] J. Huang, J.-C. Wang, J. B. L. Smith, X. Song, and Y. Wang, “Modeling the Compatibility of Stem Tracks to Generate Music Mashups,” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 1, pp. 187–195, May 2021.

[9] K. Seyerlehner, G. Widmer, and T. Pohle, “Fusing Block-level Features for Music Similarity Estimation,” in *Proceedings of the 13th International Conference on Digital Audio Effects (DAFx-10)*. DAFx, 2010.

[10] X. Hu, J. Downie, and A. Ehmann, “Lyric Text Mining in Music Mood Classification,” in *Proceedings of the 10th International Society for Music Information Retrieval Conference*. ISMIR, Jan. 2009, pp. 411–416.

[11] L. Vande Veire and T. De Bie, “From raw audio to a seamless mix: creating an automated DJ system for Drum and Bass,” *EURASIP Journal on Audio, Speech, and Music Processing*, vol. 2018, no. 134, pp. 1–21, Sep. 2018.

[12] M. Heydari, F. Cwitkowitz, and Z. Duan, “BeatNet: CRNN and Particle Filtering for Online Joint Beat Downbeat and Meter Tracking,” in *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. ISMIR, 2021.

[13] Y. Xiang, S. Gubian, B. Suomela, and J. Hoeng, “Generalized Simulated Annealing for Global Optimization: The GenSA Package,” *The R Journal*, vol. 5, no. 1, pp. 13–28, 2013.

[14] P. Knees, Á. Faraldo, P. Herrera, R. Vogl, S. Böck, F. Hörschläger, and M. L. Goff, “Two Data Sets for Tempo Estimation and Key Detection in Electronic Dance Music Annotated from User Corrections.” in *Proceedings of the 16th International Society for Music Information Retrieval Conference*. ISMIR, Sep. 2018, pp. 364–370.

[15] H. Schreiber and M. Müller, "A Crowdsourced Experiment for Tempo Estimation of Electronic Dance Music," in *Proceedings of the 19th International Society for Music Information Retrieval Conference*. ISMIR, 2018.

[16] S. Böck and M. Davies, "Deconstruct, analyse, reconstruct: How to improve tempo, beat, and downbeat estimation," in *Proceedings of the 21st International Society for Music Information Retrieval Conference*. ISMIR, Nov. 2020, pp. 574–582.

[17] F. Gouyon, A. Klapuri, S. Dixon, M. Alonso, G. Tzanetakis, C. Uhle, and P. Cano, "An experimental comparison of audio tempo induction algorithms," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1832–1844, Sep. 2006.

[18] J. Serrà, M. Müller, P. Grosche, and J. L. Arcos, "Unsupervised Music Structure Annotation by Time Series Structure Features and Segment Similarity," *IEEE Transactions on Multimedia*, vol. 16, no. 5, pp. 1229–1240, Aug. 2014.

[19] O. Nieto and J. P. Bello, "Music segment similarity using 2D-Fourier Magnitude Coefficients," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Florence, Italy: IEEE, May 2014, pp. 664–668.

[20] C. Southall, R. Stables, and J. Hockman, "Automatic Drum Transcription Using Bi-Directional Recurrent Neural Networks." in *Proceedings of the 17th International Society for Music Information Retrieval Conference*. ISMIR, Sep. 2016, pp. 591–597.

[21] ——, "Automatic drum transcription for polyphonic recordings using soft attention mechanisms and convolutional neural networks," in *Proceedings of the 18th International Society for Music Information Retrieval Conference*. ISMIR, 2017.

[22] B. Rocha, N. Bogaards, and A. Honingh, "Segmentation and timbre- and rhythm-similarity in Electronic Dance Music," University of Amsterdam, Elephantcandy, Tech. Rep., Apr. 2013.

[23] M. Panteli, B. Rocha, N. Bogaards, and A. Honingh, "A model for rhythm and timbre similarity in electronic dance music," *Musicae Scientiae*, vol. 21, no. 3, pp. 338–361, Sep. 2017.

[24] I. Sha'ath, "Estimation of key in digital music recordings, MSc Computer Science Project Report," Master's thesis, Birkbeck College, University of London, 2011.

[25] T. Royer, "Pitch-shifting algorithm design and applications in music," Master's thesis, KTH Royal Institute of Technology, School of Electrical Engineering and Computer Science, 2019.

[26] P. Knees and M. Schedl, *Music Similarity and Retrieval*, ser. The Information Retrieval Series. Springer Berlin Heidelberg, 2016, vol. 36.

[27] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Nov. 2019, pp. 3982–3992.

[28] J. Steventon, *DJing For Dummies*, 2nd ed. For Dummies, Sep. 2010.

[29] S. Rouard, F. Massa, and A. Défossez, "Hybrid transformers for music source separation," in *2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023.

[30] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Music Source Separation in the Waveform Domain," Apr. 2021, arXiv preprint arXiv:1911.13254.

[31] A. Flexer and D. Schnitzer, "Effects of album and artist filters in audio similarity computed for very large music databases," *Computer Music Journal*, vol. 34, no. 3, p. 20–28, sep 2010. [Online]. Available: https://doi.org/10.1162/COMJ_a_00004

[32] I. Vatolkin, G. Rudolph, and C. Weihs, "Evaluation of Album Effect for Feature Selection in Music Genre Recognition." in *Proceedings of the 16th International Society for Music Information Retrieval Conference*. ISMIR, Sep. 2018, pp. 169–175. [Online]. Available: https://doi.org/10.5281/zenodo.1416328

[33] A. Flexer, M. Dörfler, J. Schlüter, and T. Grill, "Hubness as a case of technical algorithmic bias in music recommendation," in *Proceedings of the 2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2018, pp. 1062–1069.

[34] A. Holzapfel, B. L. Sturm, and M. Coeckelbergh, "Ethical dimensions of music information retrieval technology," *Transactions of the International Society for Music Information Retrieval*, Sep 2018.