We first perform onset detection based on the High Frequency Content (HFC) method [31, 32]. HFC emphasizes the energy variation that occurs in the upper part of the spectrum which is typical of onsets [33]. We then select the maximum peak for each of the detected onsets.

The average peak level $\mathcal{P}_{\mu}^{(k)}$ and peak level standard deviation $\mathcal{P}_{\sigma}^{(k)}$ are defined as $\frac{1}{N}\sum_{i}^{N}\mu_{i}^{(k)}$ and $\frac{1}{N}\sum_{i}^{N}\sigma_{i}^{(k)}$, respectively. Where $\mu^{(k)}$ and $\sigma^{(k)}$ are the channel mean peak level and standard deviation in dB, which for a robust estimate are calculated from the top 75th percentile of detected peaks.

DRC normalization consists in upper bounding the peak levels of the audio. Thus, if $\mu^{(k)} > (\mathcal{P}_{\mu}^{(k)} + \mathcal{P}_{\sigma}^{(k)})$, we apply a compressor to the input audio by performing an incremental grid search of the *ratio* and *threshold* parameters until $\mu^{(k)} < (\mathcal{P}_{\mu}^{(k)} + \mathcal{P}_{\sigma}^{(k)})$. *Threshold* is the level above which compression starts, and *ratio* determines the amount of compression [29]. This is done with fixed *attack* and *release* values, i.e. the start and stop timing settings.

Artificial Reverberation—Due to the inherent characteristics of reverberation [29], an attempt to similarly normalize this effect is not trivial. Blind estimation of reverberation features, such as reverberation time (RT) or direct-to-reverberant ratio, is an open research area in itself, and considering that most of the research has been done for speech signals [34–36], its application to music is beyond the scope of this paper.

We propose instead a data augmentation approach where we stochastically add reverberation to already reverberated stems. Reverberation is added using a mixing method called the 'Abbey Road reverb trick' [37], where a chain of EQ and reverb effects is applied as a send effect, i.e. a copy of the input is processed and added to the input. Naively adding reverberation directly to the input audio has the potential to clutter the low and low-mid frequencies. In this manner, the process of learning how much reverberation is required for a given mix is carried out by the network by learning to filter out the additional reverberation that is present in the input stems.

2.2 Data preprocessing

Since most mixing effects are interrelated, applying the normalization methods separately yields inaccurate results, e.g. EQ normalization modifies the dynamics of the audio, thus modifying DRC features. To minimize this, we perform effect preprocessing progressively and based in the following order: EQ, DRC, panning and loudness. Since EQ and DRC normalization are done on a per channel basis, panning normalization is applied after the above to avoid further modification of the stereo features.

Thus, for each stem type k, average feature computation corresponds to 1) calculate $\mathcal{F}(\omega)^{(k)}$ and EQ normalize, 2) calculate $\mathcal{P}_{\mu}^{(k)}$ and $\mathcal{P}_{\sigma}^{(k)}$ and DRC normalize, 3) calculate $\mathcal{S}(\omega)^{(k)}$ and panning normalize, and 4) calculate $\mathcal{L}^{(k)}$ and loudness normalize.

We apply our reverberation preprocessing method after all the average features have been calculated. Considering that this procedure is based on a stochastic data augmentation procedure, for consistency we treat the added reverberation as noise for the normalization pipeline. The final data preprocessing method corresponds to applying reverberation augmentation followed by EQ, DRC, panning, and loudness normalization methods.

2.3 Architecture

We propose a new architecture based on [38] and [39]. It operates in the time-domain and processes raw waveforms in a frame-wise manner. The model can be divided into three parts: adaptive front-end, latent-space mixer and synthesis back-end. The model is depicted in Figure 2 and its architecture is summarized in Table 1.

The **adaptive front-end** is exactly the same as in [38], with the only difference that now the input $\hat{x}$ corresponds to K stereo tracks of length A. In the front-end, time-domain convolutions are applied to the input audio in order to learn a latent representation $\boldsymbol{Z}$ and a filter bank which output feature map $\boldsymbol{X}_1$ corresponds to a frequency band decomposition of $\hat{x}$. The **latent-space mixer** corresponds to the temporal dilated convolutions (TCN) separator block of [39], and to improve long-term dependencies learning, the TCN is followed by stacked Bidirectional Long Short-Term Memory (BLSTM) layers. Contrary to [39], the objective of the mixer is not to learn a mask for source separation, but rather to learn a mixing mask $\hat{\boldsymbol{Z}}$.

The **synthesis back-end** uses $\hat{\boldsymbol{Z}}$ to modify $\boldsymbol{X}_1$ based on the given mixing task and its design is motivated by [38]. It upsamples the mixing mask using nearest neighbor interpolation and via an element-wise multiplication applies it to each filter-bank channel source of $\boldsymbol{X}_1$ at each time step. We hypothesize that this frequency-based transformation is akin to the model learning and applying a dynamic equalization effect [40] while also filtering out the extra reverberant content of the normalized input stems.

The resulting feature map $\boldsymbol{X}_4$ is further modified via a Squeeze-and-Excitation (SE) block [41] implemented as shown in [38]. The SE layers scale the channel-wise information of $\boldsymbol{X}_4$ by applying an adaptive gain $\boldsymbol{s}_c$, and consequently, learning a loudness gain and panning transformation for each filter bank channel. The modified frequency decomposition $\boldsymbol{X}_5$ is then reconstructed using a non-trainable transposed convolution as shown in [38]. Finally, the resulting $2K$ stereo tracks are channel-wise summed into a stereo output and a hyperbolic tangent is used to avoid clipping. All convolutions use a stride of 1 to avoid ringing and filtering artifacts [42]. BLSTMs and SE layers are applied to the filter dimension.

2.4 Loss function

Based on the stereo-invariant loss function introduced by [10], we explore two variations of such loss. First, due to the perceptual nature of the mixing task and motivated

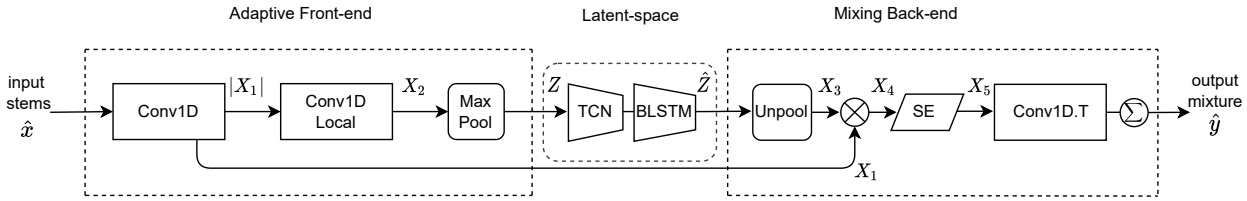

Figure 2: Block diagram of the proposed model

Table 1: Summarized architecture of the proposed model.

Layer	Output shape	Output
Input stems	$(2K, A)$	$\hat{\boldsymbol{x}}$
Conv1D	(N, A)	$\boldsymbol{X}_1$
Conv1D-Local	(N, A)	$\boldsymbol{X}_2$
MaxPooling	$(N, A/64)$	$\boldsymbol{Z}$
TCN	$(N, A/64)$	.
BLSTM	$(A/64, N)$	$\hat{\boldsymbol{Z}}$
Unpooling	(N, A)	$\boldsymbol{X}_3$
$\boldsymbol{X}_3 \times \boldsymbol{X}_1$	(N, A)	$\boldsymbol{X}_4$
SE (Abs)	(A, N)	.
SE (Global Avg)	$(1, N)$	.
SE (FC)	$(1, 16N)$	.
SE (FC)	$(1, N)$	$\boldsymbol{s}_c$
$\boldsymbol{X}_4 \times s_c$	(N, A)	$\boldsymbol{X}_5$
Conv1D.T	$(2K, A)$	.
Summation	$(2, A)$	$\hat{y}$

by [43], we apply A-weighting pre-emphasis and low-pass FIR filters (ρ) to the target and output audio frames y and $\hat{y}$, respectively. Then we compute the sum and difference signals y_{sum}=$\rho(y_{\text{L}})$+$\rho(y_{\text{R}})$ and y_{diff}=$\rho(y_{\text{L}})$-$\rho(y_{\text{R}})$.

The first loss follows closely [10] and is based on the Spectral Convergence (SC), magnitude-normalized Frobenius norm, and the L1-norm spectral log-magnitude (L1Log) as

$$\begin{aligned} L_{\text{a}} = l_{\text{SC}}(Y_{\text{sum}}, \hat{Y}_{\text{sum}}) + l_{\text{L1Log}}(Y_{\text{sum}}, \hat{Y}_{\text{sum}}) \\ + l_{\text{SC}}(Y_{\text{diff}}, \hat{Y}_{\text{diff}}) + l_{\text{L1Log}}(Y_{\text{diff}}, \hat{Y}_{\text{diff}}), \end{aligned} \tag{7}$$

where Y and $\hat{Y}$ are the respective 4096-Fast Fourier Transform (FFT) magnitude with a 25% hop size.

The second loss replaces the SC loss component with a less penalizing normalization such as the widely used L2-norm on the spectral magnitude (L2), defined as

$$\begin{aligned} L_{\text{b}} = l_{\text{L2}}(Y_{\text{sum}}, \hat{Y}_{\text{sum}}) + l_{\text{L1Log}}(Y_{\text{sum}}, \hat{Y}_{\text{sum}}) \\ + l_{\text{L2}}(Y_{\text{diff}}, \hat{Y}_{\text{diff}}) + l_{\text{L1Log}}(Y_{\text{diff}}, \hat{Y}_{\text{diff}}). \end{aligned} \tag{8}$$

We empirically found the perceptual-based pre-emphasis filter as vital when modeling a highly perceptual task such as music mixing. We also found an easier convergence during training when using a single frame-size loss rather than a multi-resolution magnitude loss.

3. EXPERIMENTS

3.1 Dataset

We first conduct experiments with a small dataset (S) which corresponds to the MUSDB18 dataset [14]. This dataset consists of wet stems for *vocals*, *drums*, *bass* and *other* (K=4), where the *mixture* is the summation of such stems. There are a total of 150 songs, of which 86 are used for training and 14 and 50 for validation and testing purposes, respectively.

We also use a private large dataset (L) for music separation which corresponds to 1,505 extra multitrack songs created in the same manner as MUSDB18. We incorporate the MUSDB18 training and validation sets into the large dataset, thus obtaining a total of 1,455+86=1,541 training songs and 50+14=64 validation songs. In both datasets, most of the songs correspond to mainstream western music, with rock and pop as the predominant genres.

To validate our method, we use a private set of 18 dry multitrack songs, where all songs have been produced by different musicians and mixing engineers. From this data we grouped the respective dry stems without applying any effect in the process.

3.2 Dataset preprocessing

We apply our preprocessing methods to both datasets independently. EQ preprocessing is computed with a 65,536 point STFT with hop size of 25% and a FIR filter of 1,001 taps. In order to avoid clipping, all audio stem channels are loudness normalized to -30 dBFS prior to the EQ feature computation and normalization.

For DRC preprocessing, to determine the timing settings of the compressor used during the normalization, we averaged the values found in mixing engineering best practices [8, 44]. *Attack* values correspond to 7.5, 10, 10 and 15 ms and *release* values to 400, 180, 500 and 666 ms, for *vocals*, *drums*, *bass* and *other*, respectively. For the incremental grid search we use *ratio* and *threshold* values within {4, 20} and {-40, -10} db respectively. For the HFC computation we use 128 mel bands of all stems with the exception of bass where we found better onset detection with 16 mel bands. Prior to the DRC preprocessing, all channels are peak normalized to -10 dB.

Panning preprocessing is done with a 2,048 point STFT with hop size of 50%. For all stems, frequency bins are re-panned until 16 kHz, with the exception of *drums*, which is re-panned until 16 kHz in order to avoid artifacts.

Reverberation augmentation is done by applying uniform random sampling on a set of 130 different impulse responses (IR) whose RT is within {2, 4} seconds (s). The

EQ used prior to the reverberation corresponds to a low-shelf and high-shelf with a fixed gain of -30 dB whose cut-off frequency is uniformly sampled within {500, 700} Hz and {7, 10} kHz, respectively. At inference, to simulate the reverberant characteristics of the training data, before applying the aforementioned augmentation, a "pre-reverb" is added in the same manner but from a different set of 400 IRs whose RT is within {1, 1.5}-*s*. A shorter RT is chosen, as high reverberation levels has been shown to have a more detrimental effect on subjective preference than low levels [45]. Reverberation augmentation is applied only to *vocals* and *other* stems. Reverberation and DRC effects are implemented using the Python package from [10].

3.3 Hyperparameters

The convolutional layers on the front-end have N=128 filters of size 64 and 128 respectively, and a 64-point window for max-pooling. In the mixer, the bottleneck layer has 256 channels, the skip connection paths have 64 channels, and the convolutional blocks have 128 channels of kernel size 3. We use 4 stacks of 6 convolutional layers with a dilation factor of 1,2,...,32. We stacked 3 BLSTMs with a hidden feature map of 64 channels. The FC layers in the SE block have 2048 ($16N$) and 128 channels respectively.

The input consists of $2K$ channels, each channel consisting of A=10-*s* audio frames at 44.1 kHz. The receptive field of the mixer is 505 samples, and taking into account the pooling operation, the overall network has a receptive field of 32,446 samples. Thus, the loss function is only calculated on a 8.52-*s* frame centered in the middle of the 10-*s* input. The network has in total 2.7M parameters. As a baseline, we use the modified *Wave-U-Net* (WUN) introduced by [9]. The same settings are used as in the original work which yields a network of 2.5M parameters.

3.4 Training

Regarding on-the-fly data augmentation, we apply a randomization of the order of the stereo channels, this is done consistently across the stems and the target mix. For the proposed network we use the pretraining from [38]. We train both models using a batch size of 4, an initial learning rate β=0.001, and the following learning rate schedule; β for 300 epochs, β/3 for 100 epochs, β/10 for 100 epochs, β/30 for 50 epochs, β/100 for 50 epochs and β/1000 for 25 epochs. An epoch consists of 1600 batches, L2 norm of the gradient is clipped by 0.2 and we use 10^{-7} for weight decay regularization. We select the model with the lowest validation loss.

4. RESULTS & ANALYSIS

We trained both networks with the preprocessed datasets S and L and the loss functions L_a and L_b, which yielded Ours-S-L$_a$, Ours-S-L$_b$, Ours-L-L$_a$ and Ours-L-L$_b$ for our proposed network, and WUN-S-L$_b$ and WUN-L-L$_b$ for Wave-U-Net. Convergence during training for WUN with L_a was unsuccessful, therefore it is excluded from the results.

	dry test set				MUSDB18 test set			
model	spectral	panning	dynamic	loudness	spectral	panning	dynamic	loudness
Normalized	<u>0.435</u>	0.593	0.168	0.633	<u>0.256</u>	0.783	0.109	0.643
WUN-S-L$_b$	0.527	0.215	0.082	0.094	**0.250**	0.250	<u>0.078</u>	0.097
Ours-S-L$_a$	0.547	0.201	0.062	<u>0.056</u>	0.299	<u>0.191</u>	0.086	0.095
Ours-S-L$_b$	**0.427**	0.207	0.063	0.061	0.276	0.212	0.085	0.084
WUN-L-L$_b$	0.551	<u>0.182</u>	0.066	**0.054**	0.279	0.195	**0.074**	**0.072**
Ours-L-L$_a$	0.590	0.191	**0.055**	0.091	0.312	0.593	0.168	0.105
Ours-L-L$_b$	0.519	**0.170**	<u>0.056</u>	0.061	0.276	**0.160**	0.084	<u>0.079</u>

Table 2: Objective metrics correspond to the average mean absolute percentage error by feature subgroup.

4.1 Quantitative evaluation

To measure how close the output mixes are to the reference mixes, we use the following audio features, spectral: centroid, bandwidth, contrast, flatness, and roll-off [46]; panning: the Panning Root Mean Square (RMS) [27]; dynamic: RMS level, dynamic spread and crest factor [47]; and LUFS loudness level [21]. All features are computed using a running mean of 0.5-*s* [27]. Table 2 shows the results for the dry test set and the MUSDB18 test set. As expected, the overall error values are larger for the dry test set. Also, the loudness, dynamics, and panning values show a closer match to the reference mix when compared to the Normalized mix, which is the sum of input stems after data preprocessing. Spectral values do not match in the same way, which could indicate that the generated mixes deviate in terms of EQ and reverberation.

4.2 Listening Test

We designed a the listening test using the Web Audio Evaluation Tool [48] and the APE interface [49]. The test is intended for professional mixing engineers only and we ask participants to rate the mixes based on Production Value, Clarity and Excitement as shown in [50]. Perceptual tests for mixing systems [9,10] often differ from MUSHRA [51] tests, as the reference sample is omitted in order to encourage a direct comparison between mixes. However, based on feedback from pilot tests, we decided to include the 4 dry stems in the test as references. In this way, the ratings may reflect the quality of transformation on each stem as applied by the mixing systems.

Fourteen participants with an average mixing engineering experience of 11.6 years took part in the test. In total there were six different songs and for each song six different mixes were presented. Each mix was 25-*s* taken from the chorus-to-verse transitions of the full mixes. The six mixes correspond to the Ours-S-L$_b$, Ours-L-L$_a$, Ours-L-L$_b$, WUN-S-L$_b$ and WUN-L-L$_b$ models plus a professional Human mix. Ours-S-L$_a$ was omitted from the test to allow more songs to be tested while avoiding listening fatigue [52]. A low-anchor was not used as it has been shown to compress the other ratings at the higher end [1]. All mixes were loudness normalized to -23 dBFS [53].

Figure 3 shows the results of the listening test and Table 3 shows the pairwise comparisons of mixes. For Production Value, there is no statistically significant difference between the Human mixtures and the models that were

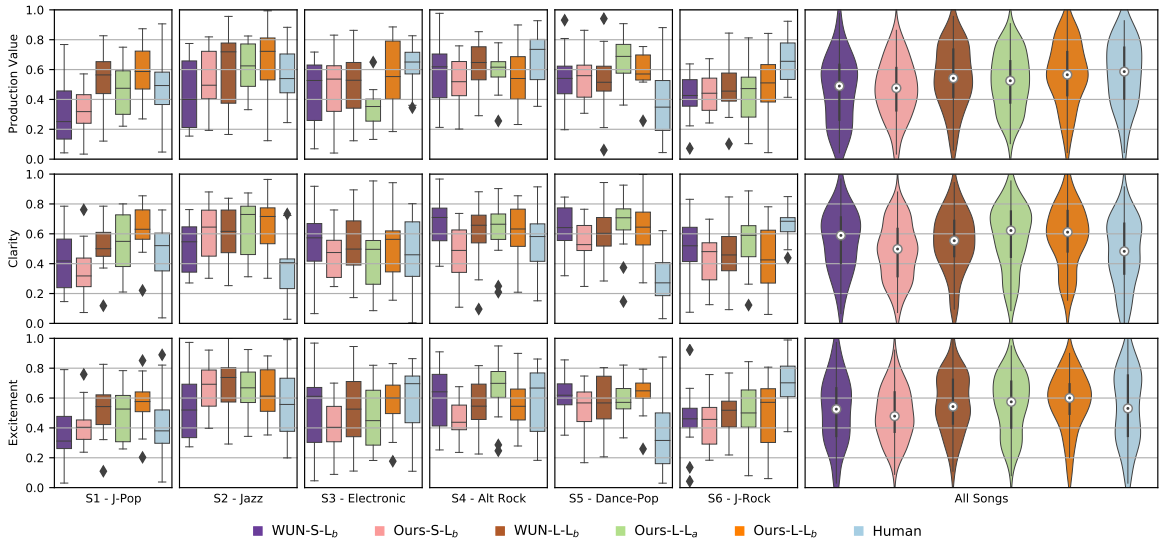

Figure 3: Listening test boxplots and violin plots for all individual six songs (S1 to S6) and all songs, respectively.

trained on the large dataset. For Clarity, Human mixes are rated lower than Ours-L-L$_a$, Ours-L-L$_b$ with a p-value of 0.008 and 0.013, respectively, indicating that our models generate mixtures with less masking. For Excitement, the null hypothesis is accepted, and therefore Human mixes are considered no different than any other mixture system. Although Ours-L-L$_b$ was among the best among all criteria, the null hypothesis is also accepted when compared to WUN-L-L$_b$. Thus, a further analysis is needed among both networks, e.g. in terms of long-term learned dependencies and ability to generate full-length coherent mixes.

In general, models trained with large datasets received the highest scores, which could confirm that lack of data has been the bottleneck of data-driven mixing systems. However, for Clarity and Excitement, the mixes by WUN-S-L$_b$ are not considered different from the rest, which might indicate that our data preprocessing method is both effective for small and large datasets.

It should be noted that, in general, none of the mixes is consistently considered as very good. This relates with other findings, where even commercial mixes made by renown engineers are not rated as high [1, 44]. This opens up a new research direction for evaluating mixing systems, as this type of test is too difficult for inexperienced participants, and when experienced participants do participate, they tend not to rate any mix as very good.

For individual songs, Human mixes have lower rates for Jazz and Dance-Pop. For the latter, low ratings on all criteria may be due to highly compressed drums. However, the high ratings in terms of Excitement and Production Value for the Human J-Rock mix might be due to hard-panned guitars. In contrast, the models are conservative when it comes to panning and are unlikely to hard-pan sources, however a further analysis is required. Furthermore, although we show the models successfully mixing multiple genres, an in-depth analysis of the genre distribution of the dataset with ratings by genre is needed.

Production Value	WUN-S-L$_b$	Ours-S-L$_b$	WUN-L-L$_b$	Ours-L-L$_a$	Ours-L-L$_b$	Human
WUN-S-L$_b$	·	o	$\underline{\star}$	o	$\underline{**}$	$\underline{\star}$
Ours-S-L$_b$	o	·	o	o	$\underline{**}$	$\underline{\star}$
WUN-L-L$_b$	$\overline{\star}$	o	·	o	o	o
Ours-L-L$_a$	o	o	o	·	o	o
Ours-L-L$_b$	$\overline{**}$	$\overline{**}$	o	o	·	o
Human	$\overline{\star}$	$\overline{\star}$	o	o	o	·
Clarity	**WUN-S-L$_b$**	**Ours-S-L$_b$**	**WUN-L-L$_b$**	**Ours-L-L$_a$**	**Ours-L-L$_b$**	**Human**
WUN-S-L$_b$	·	o	o	o	o	o
Ours-S-L$_b$	o	·	o	$\underline{\star}$	$\underline{**}$	o
WUN-L-L$_b$	o	o	·	o	o	o
Ours-L-L$_a$	o	$\overline{\star}$	o	·	o	$\overline{\star}$
Ours-L-L$_b$	o	$\overline{**}$	o	o	·	$\overline{\star}$
Human	o	o	o	$\underline{\star}$	$\underline{\star}$	·
Excitement	**WUN-S-L$_b$**	**Ours-S-L$_b$**	**WUN-L-L$_b$**	**Ours-L-L$_a$**	**Ours-L-L$_b$**	**Human**
WUN-S-L$_b$	·	o	o	o	o	o
Ours-S-L$_b$	o	·	$\underline{*}$	$\underline{\star}$	$\underline{*}$	o
WUN-L-L$_b$	o	$\overline{*}$	·	o	o	o
Ours-L-L$_a$	o	$\overline{\star}$	o	·	o	o
Ours-L-L$_b$	o	$\overline{*}$	o	o	·	o
Human	o	o	o	o	o	·

Table 3: Post hoc Mann-Whitney test results of pairwise comparison with Bonferroni Correction. o > 0.05, $\star < 0.05$, $* < 0.01$, $** < 0.001$. E.g. when y-axis is compared to x-axis, $\overline{*}$ or $\underline{*}$ indicate y-axis is significantly better or worse than x-axis for a p-value < 0.01, respectively.

5. CONCLUSION

We present a novel data preprocessing approach that allows us to train deep learning networks with existing wet or processed multitrack data by reusing them to perform an automatic mixing task. During inference, we apply the same data preprocesing to dry multitrack data and we tested the generated mixes via objective and subjective tests. We introduced a new deep learning architecture designed for the proposed task, a perceptual-based loss function along with a redesigned listening test aimed at experienced mixing engineers. The results indicate that our approach successfully achieves automatic loudness, EQ, DRC, panning, and reverberation music mixing. Resulting mixes compared to professional mixes scored higher in terms of Clarity and are indistinguishable in terms of Production Value and Excitement. We believe that the proposed preprocessing can be applied to other data-driven MIR tasks.

6. ACKNOWLEDGMENTS

We would like to express special thanks to K. Gokan and S. Masui for their valuable comments and all the mixing engineers from Sony Music Entertainment Japan, Sony Europe and Queen Mary University of London who participated in the listening test.

7. REFERENCES

[1] R. Stables, J. D. Reiss, and B. De Man, *Intelligent Music Production*. Focal Press, 2019.

[2] P. D. Pestana and J. D. Reiss, "Intelligent audio production strategies informed by best practices," in *53rd Conference on Semantic Audio: Audio Engineering Society*, January 2014.

[3] B. De Man, J. D. Reiss, and R. Stables, "Ten years of automatic mixing," in *3rd AES Workshop on Intelligent Music Production*, September 2017.

[4] G. Bocko, M. F. Bocko, D. Headlam, J. Lundberg, and G. Ren, "Automatic music production system employing probabilistic expert systems," in *Audio Engineering Society Convention 129*. Audio Engineering Society, November 2010.

[5] B. De Man and J. D. Reiss, "A knowledge-engineered autonomous mixing system," in *Audio Engineering Society Convention 135*. Audio Engineering Society, October 2013.

[6] F. Everardo, "Towards an automated multitrack mixing tool using answer set programming," in *Proceedings of the 14th Sound and Music Computing Conference*, Espoo, Finland, July 2017.

[7] J. D. Reiss, "Intelligent systems for mixing multichannel audio," in *17th International Conference on Digital Signal Processing*. IEEE, 2011, pp. 1–6.

[8] P. Pestana, "Automatic mixing systems using adaptive digital audio effects," Ph.D. dissertation, Universidade Católica Portuguesa, 2013.

[9] M. A. Martínez-Ramírez, D. Stoller, and D. Moffat, "A deep learning approach to intelligent drum mixing with the wave-u-net." Journal of the Audio Engineering Society, 2021.

[10] C. J. Steinmetz, J. Pons, S. Pascual, and J. Serrà, "Automatic multitrack mixing with a differentiable mixing console of neural audio effects," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2021.

[11] D. Moffat and M. B. Sandler, "Approaches in intelligent music production," *Arts*, vol. 8, no. 5, p. 14, September 2019. [Online]. Available: https://doi.org/10.3390/arts8040125

[12] T. Wilmering, D. Moffat, A. Milo, and M. B. Sandler, "A history of audio effects," *Applied Sciences*, vol. 10, no. 3, p. 791, January 2020. [Online]. Available: https://doi.org/10.3390/app10030791

[13] Y. Mitsufuji, G. Fabbro, S. Uhlich, F.-R. Stöter, A. Défossez, M. Kim, W. Choi, C.-Y. Yu, and K.-W. Cheuk, "Music demixing challenge 2021," *Frontiers in Signal Processing*, vol. 1, 2022. [Online]. Available: https://doi.org/10.3389/frsip.2021.808395

[14] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "The MUSDB18 corpus for music separation," Dec. 2017. [Online]. Available: https://doi.org/10.5281/zenodo.1117372

[15] D. Barchiesi and J. Reiss, "Reverse engineering of a mix," *Journal of the Audio Engineering Society*, vol. 58, no. 7/8, pp. 563–576, 2010.

[16] D. Moffat and M. B. Sandler, "Automatic mixing level balancing enhanced through source interference identification," in *Audio Engineering Society Convention 146*, Dublin, Ireland, March 2019.

[17] J. T. Colonel and J. Reiss, "Reverse engineering of a recording mix with differentiable digital signal processing," *The Journal of the Acoustical Society of America*, vol. 150, no. 1, pp. 608–619, 2021.

[18] J. Imort, G. Fabbro, M. A. Martínez-Ramírez, S. Uhlich, Y. Koyama, and Y. Mitsufuji, "Distortion audio effects: Learning how to recover the clean signal," in *23rd International Society for Music Information Retrieval Conference (ISMIR)*, 2022.

[19] J. Koo, S. Paik, and K. Lee, "Reverb conversion of mixed vocal tracks using an end-to-end convolutional deep neural network," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[20] D. Singh and B. Singh, "Investigating the impact of data normalization on classification performance," *Applied Soft Computing*, vol. 97, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494619302947

[21] R. ITU-R, "ITU-R BS. 1770-2, algorithms to measure audio programme loudness and true-peak audio level," *International Telecommunications Union, Geneva*, 2011.

[22] C. J. Steinmetz and J. D. Reiss, "pyloudnorm: A simple yet flexible loudness meter in python," in *Audio Engineering Society Convention 150*, 2021.

[23] F. G. Germain, G. J. Mysore, and T. Fujioka, "Equalization matching of speech recordings in real-world environments," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2016.

[24] R. W. Schafer, "What is a savitzky-golay filter?" *IEEE Signal processing magazine*, vol. 28, no. 4, pp. 111–117, 2011.

[25] P. Gaydecki, *Foundations of digital signal processing: theory, algorithms and hardware design*. Iet, 2004, vol. 15.

[26] J. O. Smith III, *Introduction to Digital Filters: with Audio Applications*. W3K Publishing, 2007, vol. 2.

[27] G. Tzanetakis, R. Jones, and K. McNally, "Stereo panning features for classifying recording production style." in *8th International Society for Music Information Retrieval Conference (ISMIR)*, 2007.

[28] C. Avendano, "Frequency-domain source identification and manipulation in stereo mixes for enhancement, suppression and re-panning applications," in *IEEE Workshop on Applications of Signal Processing to Audio and Acoustics*, 2003.

[29] U. Zölzer *et al.*, *DAFX-Digital audio effects*. John Wiley & Sons, 2002.

[30] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler, "A tutorial on onset detection in music signals," *IEEE Transactions on speech and audio processing*, vol. 13, no. 5, pp. 1035–1047, 2005.

[31] P. Masri, "Computer modelling of sound for transformation and synthesis of musical signals," Ph.D. dissertation, University of Bristol, 1996.

[32] P. Brossier, Tintamar, E. Müller, N. Philippsen, T. Seaver, H. Fritz, cyclopsian, S. Alexander, J. Williams, J. Cowgill, and A. Cruz, "Aubio - a library for audio and music analysis," February 2019. [Online]. Available: https://doi.org/10.5281/zenodo.2578765

[33] P. Brossier, "Automatic annotation of musical audio for interactive applications," Ph.D. dissertation, Queen Mary University of London, 2006.

[34] J. Eaton, N. D. Gaubitch, A. H. Moore, and P. A. Naylor, "Estimation of room acoustic parameters: The ace challenge," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 24, no. 10, pp. 1681–1693, 2016.

[35] H. Löllmann, E. Yilmaz, M. Jeub, and P. Vary, "An improved algorithm for blind reverberation time estimation," in *Proceedings of International Workshop on Acoustic Echo and Noise Control (IWAENC)*, 2010, pp. 1–4.

[36] S. Duangpummet, J. Karnjana, W. Kongprawechnon, and M. Unoki, "Blind estimation of room acoustic parameters and speech transmission index using mtf-based cnns," in *29th European Signal Processing Conference (EUSIPCO)*, 2021, pp. 181–185.

[37] P. White, "Processing reverb," *Sound On Sound*, vol. November 2020 Edition, 2022. [Online]. Available: https://www.soundonsound.com/techniques/processing-reverb

[38] M. A. Martínez-Ramírez, E. Benetos, and J. D. Reiss, "Deep learning for black-box modeling of audio effects," *Applied Sciences*, vol. 10, no. 2, p. 638, 2020. [Online]. Available: https://doi.org/10.3390/app10020638

[39] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM transactions on audio, speech, and language processing*, vol. 27, no. 8, pp. 1256–1266, 2019.

[40] V. Välimäki and J. D. Reiss, "All about audio equalization: Solutions and frontiers," *Applied Sciences*, vol. 6, no. 5, p. 129, 2016.

[41] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[42] J. Pons, S. Pascual, G. Cengarle, and J. Serrà, "Upsampling artifacts in neural audio synthesis," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021.

[43] A. Wright and V. Välimäki, "Perceptual loss function for neural modeling of audio systems," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020.

[44] B. De Man, "Towards a better understanding of mix engineering," Ph.D. dissertation, Queen Mary University of London, 2017.

[45] B. De Man, K. McNally, and J. D. Reiss, "Perceptual evaluation and analysis of reverberation in multitrack music production," *Journal of the Audio Engineering Society*, 2017.

[46] G. Peeters, "A large set of audio features for sound description (similarity and classification) in the cuidado project," *Analysis/Synthesis Team. IRCAM, Paris, France*, vol. 54, no. 0, pp. 1–25, 2004.

[47] Z. Ma, B. De Man, P. D. Pestana, D. A. Black, and J. D. Reiss, "Intelligent multitrack dynamic range compression," *Journal of the Audio Engineering Society*, vol. 63, no. 6, pp. 412–426, 2015.

[48] N. Jillings, B. De Man, D. Moffat, J. D. Reiss, and R. Stables, "Web audio evaluation tool: A browser-based listening test environment," in *International Sound and Music Computing Conference*, Maynooth, Ireland, July 2015.

[49] B. De Man and J. D. Reiss, "APE: Audio perceptual evaluation toolbox for matlab," in *Audio Engineering Society Convention 136*. Audio Engineering Society, 2014.

[50] P. D. Pestana and J. D. Reiss, "A cross-adaptive dynamic spectral panning technique." in *17th International Conference on Digital Audio Effects (DAFx-14)*, 2014.

[51] ITU-R BS. 1534-1, "Method for the subjective assessment of intermediate sound quality (MUSHRA)," *International Telecommunications Union, Geneva*, 2001.

[52] R. Schatz, S. Egger, and K. Masuch, "The impact of test duration on user fatigue and reliability of subjective quality ratings," *Journal of the Audio Engineering Society*, vol. 60, no. 1/2, pp. 63–73, 2012.

[53] EBU, "EBU recommendation: Loudness normalisation and permitted maximum level of audio signals," EBU-R128: Saconnex, 2014.

MUSIC-STAR: A STYLE TRANSLATION SYSTEM FOR AUDIO-BASED RE-INSTRUMENTATION

Mahshid Alinoori **Vassilios Tzerpos**
Department of Electrical Engineering and Computer Science, York University, Canada
{mahshida, bil}@yorku.ca

ABSTRACT

Music style translation aims to generate variations of existing pieces of music by altering the style-related characteristics of the original piece while content, such as the melody, remains unchanged. These alterations could involve timbre translation, re-harmonization, or music rearrangement. Previous studies have achieved promising results utilizing time-frequency and symbolic music representations. Music style translation on raw audio has also been investigated and applied to single-instrument pieces. Although processing raw audio is more challenging, it provides richer information about timbres, dynamics, and articulations.

In this paper, we introduce Music-STAR, the first audio-based translation system that translates the existing instruments in a piece into a set of target instruments without using source separation. To conduct our experiments, we also present an audio dataset that contains two-track pieces performed by two instrument sets alongside their stems. We carry out subjective and objective evaluations to compare Music-STAR with a variety of baseline methods and demonstrate its superiority.

1. INTRODUCTION

Music style translation is defined as transforming the style-variant components of a music piece to create variations that preserve the content. This can take several forms depending on how "music style" is characterized. Dai et al. [1] classify style into three categories: composition, performance, and timbre. Recent works have mainly focused on timbre translation [2–7] and composition style translation [8–12].

Timbre translation aims to alter the timbre information, which typically results in a change in instrumentation. Timbre translation models mostly use time-frequency representations [2–6]. Some of these [4–6] treat the spectrograms as images and apply GAN-based models [13,14] designed for image-to-image translation [15]. Timbre translation has also been explored by integrating classic signal processing and deep learning methods [16, 17].

The universal translation network [7] is the only model that works with audio waveforms directly. Inspired by the WaveNet autoencoder [18], it translates an arbitrary source piece to several specific timbre domains. Although one universal encoder is used to encode the domain-independent features, every domain needs to have a specific conditional WaveNet [19] decoder.

On the other hand, composition style transfer attends to tasks such as re-harmonization and music rearrangement. [1] Almost all works on composition style translation exploit symbolic representations, i.e., MIDI and piano rolls. Some of these models focus on music rearrangement by altering the accompaniments [8–10] where the style is associated with the genre. MIDI-VAE [9] is among the few cases that do not overlook the dynamics and account for note velocities. Wang et al. [11] has introduced the only GAN-based model that operates on symbolic representation to perform genre transformation. Hung et al. [12] approach music rearrangement by modifying the instrumentation, where they establish a correlation between rearrangement and transforming multiple timbres in a musical piece.

In this paper, we explore music re-instrumentation of audio waveforms that contain more than one instrument. We present several baseline solutions and then propose Music-STAR, a system designed explicitly for audio-based translation, which is built upon the WaveNet autoencoder [18]. To conduct our experiments, we also present a dataset called StarNet, in which every piece of music is performed by two different sets of instruments. The source code, audio samples, and supplementary materials are available at https://mahshidaln.github.io/Music-STAR.

2. DATASET

In order to train and evaluate Music-STAR and the baseline models, we need an audio dataset that contains multi-track pieces played with different sets of instruments alongside their stems. For this purpose, we have created the StarNet dataset, in which every piece is composed of two instrument tracks from two domains: strings-piano and clarinet-vibraphone. In other words, for every piece, the dataset includes strings-piano and clarinet-vibraphone mixtures as well as their corresponding isolated tracks.

We have chosen piano ↔ vibraphone and strings ↔ clarinet translations to preserve the type of excitation in

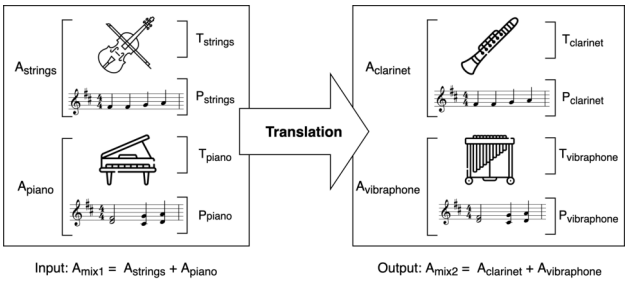

Figure 1. An example of multi-instrument translation where the timbre components of strings ($T_{strings}$) and piano (T_{piano}) are translated into clarinet ($T_{clarinet}$) and vibraphone ($T_{vibraphone}$), respectively, while retaining the corresponding pitch components ($P_{strings} = P_{clarinet}$ and $P_{piano} = P_{vibraphone}$).

the source/target instruments (discrete excitations in piano and vibraphone versus continuous excitations in clarinet and strings).

Accessing recorded music stems is often hard due to copyright limitations. Therefore, we select a variety of music pieces from MusicNet [20] and other freely available classical music MIDI collections. Instead of using their original instruments, we apply virtual instruments for the aforementioned combinations. The resulting dataset contains two domains, one for each instrument combination, adding up to roughly 11 hours of audio for each domain. The StarNet dataset is available at `https://zenodo.org/record/6917099`.

To conduct our experiments, we use two versions of StarNet:

1. Preprocessed StarNet: The preprocessed version of StarNet includes uncompressed stereo WAV files with a sample rate of 44.1 kHz and a bit depth of 16 bits. The preprocessing step includes detecting and removing the intervals where one or both instruments are silent to ensure their simultaneous presence for a considerable amount of time. After the silence removal, the dataset size shrinks to roughly 9 hours. Silence detection is done by indicating the minimum loudness and minimum silence duration.

2. Reduced StarNet: The reduced version of StarNet is obtained after resampling the preprocessed version at 16 kHz, merging the two audio channels and outputting mono audio, and finally quantizing the audio by 8-bit mu-law encoding.

3. METHODOLOGY

Pitch and timbre are among the fundamental acoustic properties of every audio track in a recorded mixture. As we address instrumentation changes in our work, timbre is regarded as the style component, while pitch constitutes the content we intend to preserve. Based on this definition, we introduce the following notation of a piano track and its style and content components:

$$A_{piano} := T_{piano} + P_{piano}$$

where A is an audio signal, T is the timbre component, and P is the set of pitch components and their embodied durations in that signal. Separating style and content components, also known as disentanglement, has been leveraged in previous music translation studies [7,9,12] and is mostly addressed by adversarial learning in encoder-decoder architectures. Note that T and P in the notations are conceptual terms, and the corresponding features will be extracted by the autoencoders and represented by the embeddings.

An example of single-instrument translation is shown below, where a piano track is translated to vibraphone in a way that the pitch component is retained, and the timbre component alters:

1. $Input = A_{piano} := T_{piano} + P_{piano}$
2. $Output = A_{vibraphone} := T_{vibraphone} + P_{piano}$

Our task is to tackle a more complex problem, i.e., dealing with multi-instrument pieces (Fig. 1). In the case of our dataset, the input audio signal will be one of the following:

1. $A_{mix1} = A_{strings} + A_{piano} :=$
   $T_{strings} + P_{strings} + T_{piano} + P_{piano}$
2. $A_{mix2} = A_{clarinet} + A_{vibraphone} :=$
   $T_{clarinet} + P_{clarinet} + T_{vibraphone} + P_{vibraphone}$

As a result, pitch-timbre disentanglement does not suffice here as we need to isolate two sets of timbre and pitch components.

In this section, we first introduce the baseline methods for performing multi-instrument translation, i.e., single-instrument translation pipeline and separation-based translation pipeline. Then we present our proposed methods, i.e., the embedding-supervised method and Music-STAR.

3.1 Single-instrument Translation Pipeline

The most simplistic approach to multi-instrument translation is to translate each instrument track separately and obtain the final output by mixing them. However, this is only possible when the music stems are available.

In order to implement this method, we employ an existing audio-based translation model [7], which is built upon the WaveNet autoencoder [18]. This model consists of a universal encoder and six decoders corresponding to six target domains, all collected from the MusicNet [20] dataset.

In this universal network, the temporal encoder maps the pitch components into an embedding space utilizing a WaveNet-like architecture. The decoders are conditioned on the pitch embeddings and generate audio samples that entail the specific timbre they have been trained for in an autoregressive manner. A domain confusion network is employed during training to ensure that no domain-specific information is included in the embeddings by imposing domain confusion loss [21]. The authors also emphasize the role of distorting the input by modulating the pitch locally