TOWARDS ROBUST MUSIC SOURCE SEPARATION ON LOUD COMMERCIAL MUSIC

Chang-Bin Jeon
Department of Intelligence and Information
Music and Audio Research Group (MARG)
Seoul National University
vinyne@snu.ac.kr

Kyogu Lee
Department of Intelligence and Information
Music and Audio Research Group (MARG)
Seoul National University
kglee@snu.ac.kr

ABSTRACT

Nowadays, commercial music has extreme loudness and heavily compressed dynamic range compared to the past. Yet, in music source separation, these characteristics have not been thoroughly considered, resulting in the domain mismatch between the laboratory and the real world. In this paper, we confirmed that this domain mismatch negatively affect the performance of the music source separation networks. To this end, we first created the out-of-domain evaluation datasets, *musdb-L* and *XL*, by mimicking the music mastering process. Then, we quantitatively verify that the performance of the state-of-the-art algorithms significantly deteriorated in our datasets. Lastly, we proposed *LimitAug* data augmentation method to reduce the domain mismatch, which utilizes an online limiter during the training data sampling process. We confirmed that it not only alleviates the performance degradation on our out-of-domain datasets, but also results in higher performance on in-domain data.

1. INTRODUCTION

Recent commercial music has extreme loudness compared to the past [1, 2]. Since many artists and producers want their music to be perceptually louder, it has become so trendy that it rather harms the quality of music [3] and even there exists the expression 'loudness war' [4].

A dynamic range compressor [5] is used to increase the loudness of music while keeping the digital level under 0 decibels relative to full scale (dBFS), which is the maximum possible level in the digital domain. It is a time-varying non-linear processor that adjusts the level of the signal when the level exceeds the threshold. Especially, a limiter refers to a dynamic range compressor that strongly compresses the signal with a high ratio parameter above 1:10 and increases the gain of the signal by the headroom obtained through compression.

In the last stage of music production, so-called mastering, engineers are often asked to increase the overall loudness of music. A limiter is a core tool for achieving it; it is used to make small parts of music to be louder and loud parts of music fit under 0 dBFS [7, 8]. In this process, original signals are distorted and become louder. The example of a limiter usage is depicted in Figure 1.

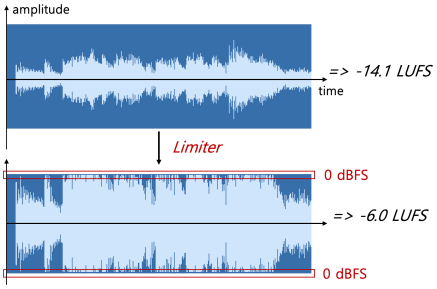

Figure 1. The short example of a limiter applied music source in our *musdb-XL* dataset. Recent commercial music has this loud volume and distorted signal characteristics.

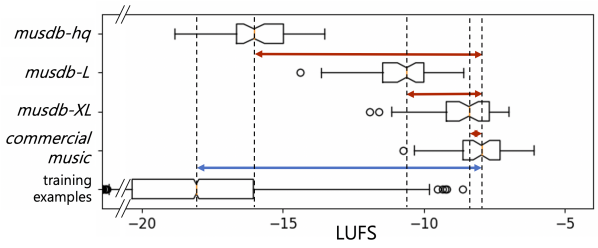

Figure 2. The boxplot representing the loudness distributions of different evaluation datasets, recent commercial music, and training examples generated from the official implementation of *Open-unmix* [6].

However, these characteristics have not yet been thoroughly considered in the music source separation. Although many data augmentation techniques have been proposed, such as random gain scaling and mixing of different instrumental stem sources [9, 10], the training examples have small overall loudness that is far from real-world commercial music [1], as can be seen in the blue line of Figure 2. This causes a huge domain shift between train and real-world domains. The term LUFS in Figure 2 is an abbreviate for loudness unit relative to full scale, which is a

[1] Based on the investigation of 50 songs in *'Pop Life playlist'* created by Tidal, May. 2022.

measure of music loudness [11, 12].

Furthermore, since the standard benchmark datasets for music source separation, *musdb* [13] and *musdb-hq* [14], do not reflect the characteristics of loud commercial music as shown in the red lines of Figure 2, they are unable to show performance degradation comes from the domain shift. Therefore, we conjecture that even good-performing networks based on *musdb* test subset may exhibit worse performance on real world. This problem is no exception to other datasets [15–17] for music source separation, since they do not consider the music mastering process.

Based on this question, we investigated how heavily compressed music and its loudness affect the degradation of music source separation networks. To this end, we manually built new evaluation datasets by applying a limiter to the *musdb-hq* test subset [14] to get loud and compressed music imitating the modern music mastering process. One is the *musdb-L* dataset, with increased loudness to an appropriate degree for pleasant sound following [3]. The other is the *musdb-XL* dataset, which raised the loudness to an excessive degree, as often found in recent popular music. Using our datasets, we confirmed that more dynamic range compression in test domains results in more performance degradation of the networks.

Moreover, we conducted various experiments on the training data creating methods to reduce the domain shift between train and real world data from the perspective of music loudness and heavy compression. We trained music source separation networks using the training examples constructed by *(1)* linear gain increasing, *(2)* the proposed *LimitAug* method which utilizes an online limiter during the sampling process of training examples, *(3)* simple input loudness normalization, and *(4)* the loudness normalization after applying the *LimitAug*. We confirmed that all of the methods not only showed robust performance on out-of-domain *musdb-L* and *musdb-XL* data, but also showed better performance on in-domain *musdb-hq* test dataset than the baseline.

To summarize, our contributions are three-fold.

- We built *musdb-L* and *XL* datasets [2], which have comparable overall loudness to commercial music, for evaluation of music source separation algorithms.
- Using *musdb-L* and *XL*, we quantitatively confirmed that the domain shift causes performance degradation of the state-of-the-art networks that were trained without considering loud and compressed music characteristics.
- We proposed *LimitAug* [3] data augmentation method and experimentally confirmed that it is beneficial to alleviate the domain shift between train data and the *musdb-L* or *XL*.

[2] https://github.com/jeonchangbin49/musdb-XL
[3] https://github.com/jeonchangbin49/LimitAug

dataset	Loudness [LUFS]			
	min	max	median	mean (std)
musdb-hq	-18.84	-13.52	-16.02	-15.92 (1.27)
musdb-L	-14.39	-8.61	-10.61	-10.89 (1.19)
musdb-XL	-11.93	-6.99	-8.41	-8.61 (1.17)
commercial	**-10.75**	**-6.10**	**-7.96**	**-8.05 (1.06)**

Table 1. Loudness statistics of *musdb-hq*, *musdb-L*, *musdb-XL* datasets, and the investigated commercial music.

2. MUSDB-L AND MUSDB-XL

musdb [13] and *musdb-hq* [14] have been the standard benchmark datasets on music source separation since their presentation in Signal Separation and Evaluation Challenge (SiSEC) 2018 [18]. They consist of various genres of 150 professionally produced songs — 100 songs for train, 50 songs for test — from folk, indie to electronic, rock genres. Each song has its constituting 4 stems, *vocals*, *drums*, *bass*, and the remaining as *other*. The *musdb-hq* dataset is the uncompressed version of *musdb* with the extended frequency bandwidth from 16kHz to 22.05kHz, which is a full bandwidth of 44.1kHz sample rate. We used *musdb-hq* for high-quality dataset construction.

Since the test subset of *musdb-hq* has small overall loudness compared to commercial mastering-finished music, we conjectured that it could not fully reflect the performance degradation related to heavy dynamic range compression, which prevails in recent commercial music. In addition, to the best of our knowledge, there is no dataset for music source separation that reflects these characteristics or considers the music mastering process. Therefore, we built *musdb-L* and *musdb-XL* datasets, which have loud and compressed characteristics similar to commercial music. *L* and *XL* each stands for *Loud* and *eXtremely Loud*.

2.1 Dataset construction

To reflect the characteristics of commercial music, we created datasets by imitating the music mastering process. Both datasets were made by manually applying the commercial digital limiter, iZotope Ozone 9 Maximizer [4], to the *musdb-hq* test subset. For each *musdb-L* and *musdb-XL*, we controlled the threshold parameter of a limiter so that compression is applied about 3-4 dB and 6-7 dB in loud parts of *mixture* tracks of the *musdb-hq*. As shown in Table 1, *musdb-L* and *musdb-XL* are about 5 and 7 LUFS louder than the original *musdb-hq* dataset on average, respectively. *musdb-L* has insufficient loudness compared to loud commercial music but less distorted sources. *musdb-XL* has comparable loudness to commercial music and more distorted sound than *musdb-L*.

To make the ground truth stem tracks of each mixture, we calculated the sample-wise ratio between the limiter applied mixture and the original mixture, then multiply it to the individual stems to make the ground truth stems for *musdb-L* and *XL*.

[4] https://www.izotope.com/en/products/ozone.html

3. METHODS

In general, a limiter is used as the last signal processor in music mastering of commercial music [7, 8]. Since it tends to be used excessively in modern commercial music [1, 2, 4], it should be considered in music source separation for the development of robust application.

We assumed that the key differences between the real-world mastering-finished music, i.e. a limiter applied music, and the standard training examples for music source separation are *i)* the overall amplitude scale, and *ii)* the signal distortion caused by a limiter.

Here we introduce two ways to avoid each domain mismatch problem.

3.1 Input loudness normalization

First of all, the simple loudness normalization, which is a linear gain adjustment of network inputs to the pre-defined reference level, is the easiest technique to avoid the domain shift caused by *i)* overall amplitude scale mismatch. This can be categorized into two, *(i)* input loudness normalization during both the training and evaluation stages of networks, and *(ii)* normalization only at the evaluation stage.

The method *(i)* is already used in various studies with different ways. For example, the network such as *Demucs v3* [19], uses input standardization in time domain based on the mean and standard deviation of the training data. In *Open-unmix* [6], the network implicitly normalizes the input by utilizing trainable input scaling parameters that works in time-frequency domain. However, we assumed that explicitly adjusting the gain of the inputs based on the waveform, thereby minimizing the overall amplitude scale mismatch between train and test domain, can greatly reduce the performance degradation comes from the domain shift.

The method *(ii)*, normalization only at the evaluation stage, can be considered as a readily applicable workaround for the models that were trained without considering music loudness. This method is a simple idea to reduce the overall amplitude scale difference between train and real-world domain. Since the music source separation networks are non-linear systems, we hypothesized that linear scale difference of the inputs might affect the quality of final outputs. That is, input normalization only at the evaluation stage can be a simple, yet effective trick for models that were already trained without considering the domain mismatch.

Though these methods can mitigate the domain mismatch related to *i)* amplitude scale, there needs to be another strategies that can reflect *ii)* the signal distortion caused by a limiter.

3.2 *LimitAug*

The best way to consider the characteristics of the limiter applied source is simply using the limiter during the network training. Therefore, we propose the *LimitAug* data augmentation method, which utilizes an online limiter during the training examples construction process, to reduce the domain shift between the training examples and real-world commercial music. The *LimitAug* can be considered as creating train examples that forcibly reflect the distortion that comes from a limiter, which cannot be reflected by the simple input loudness normalization technique.

```
# mix, tgt : mixture and tgt sources.
# gain(src, tgt_lufs) : calculate lufs of source,
#  then adjust its gain targeting given lufs,
#  return output and adjusted gain
# gain_adj(src, adj_gain) : gain adjustment
#  of source with given adj_gain.
tgt_lufs = random_sampled_tgt_lufs
mix_loud, _ = gain(mix, tgt_lufs)
mix_loud_limited = limiter(mix_loud)
ratio = mix_loud_limited / mix
tgt_loud = tgt * ratio
if input_loud_norm:
  mix_loud_limited, adj_gain = gain(src, tgt_lufs)
  tgt_loud = gain_adj(tgt_loud, adj_gain)
estimates = network(mix_loud_limited)
loss = objective(estimates, tgt_loud)
```

Figure 3. `numpy`-like pseudocode of the proposed *LimitAug* method.

The pseudocode for *LimitAug* is shown in Figure 3. First, calculate the LUFS of the mixture created by the data sampling process. Second, adjust the gain of the input mixture source targeting the randomly sampled LUFS value. Then, it is followed by the online limiter to fit the waveform under 0 dBFS. Lastly, calculate the sample-wise (A sample refers to an each waveform value) ratio between the limiter applied mixture source and the original mixture source, then multiply the ratio to the original target source to get the ground truth target signal of the limiter applied mixture source.

When adjusting the gain and applying the limiter to the input mixture, for example, if the input mixture had -15 LUFS and the randomly sampled target LUFS was -10, note that the gain-scaled mixture by +5 dB does not have exact -10 LUFS due to the compression of a limiter and the nature of LUFS calculation, which considers frequency weighting [11, 12].

The proposed *LimitAug* can be used with other data augmentation methods; in our study, random gain scaling, channel swap and mixing of different instrumental stem sources [9, 10] were used. Also, additional loudness normalization can be applied after the *LimitAug* as depicted in conditional statement of Figure 3, so that the overall amplitude scale mismatch between training and evaluation stages be minimized.

4. EXPERIMENTS

Here we briefly summarize our following experiments.

In Section 5.1, we quantitatively evaluated various music source separation networks that were trained without considering the loudness and heavy dynamic range compression, and confirmed the performance degradation on *musdb-L* and *musdb-XL*, compared to the origianl *musdb-hq* evaluation dataset.

network	extra train data	test data	SDR median (mean) [dB]				
			vocals	*bass*	*drums*	*other*	*avg*
Open-unmix [6]	-	*hq*	6.16 (2.54)	5.03 (2.67)	6.00 (5.46)	4.22 (3.46)	5.35 (3.53)
		L	6.33 (1.63)	4.81 (2.71)	5.82 (5.38)	4.11 (3.42)	5.27 (3.28)
		XL	5.98 (0.89)	4.76 (2.59)	4.97 (4.89)	4.04 (3.29)	4.94 (2.92)
TFC-TDF -U-net [20]	-	*hq*	7.18 (4.26)	5.59 (3.35)	5.76 (5.30)	4.04 (3.18)	5.64 (4.02)
		L	7.03 (3.65)	5.41 (3.08)	5.52 (5.09)	3.67 (3.00)	5.41 (3.71)
		XL	6.95 (3.14)	5.48 (2.90)	5.11 (4.68)	3.55 (2.82)	5.27 (3.39)
Demucs v3-A [19]	-	*hq*	**8.11 (5.22)**	**9.34** (6.21)	**8.57 (8.01)**	**5.51 (5.03)**	**7.88 (6.12)**
		L	7.54 (5.15)	9.32 **(6.22)**	8.26 (7.65)	5.51 (5.01)	7.66 (6.01)
		XL	7.30 (4.86)	9.19 (6.14)	7.62 (6.78)	5.37 (4.97)	7.37 (5.69)
Open-unmix [6]	✓	*hq*	7.02 (4.93)	5.91 (4.06)	7.18 (6.91)	4.94 (4.76)	6.26 (5.17)
		L	6.83 (5.12)	6.23 (4.09)	7.07 (6.92)	4.94 (4.78)	6.27 (5.23)
		XL	6.70 (4.77)	6.16 (3.87)	6.80 (6.48)	4.89 (4.61)	6.14 (4.93)
Spleeter [21]	25000+	*hq*	6.51 (4.42)	4.77 (3.57)	6.00 (6.09)	4.22 (4.12)	5.38 (4.55)
		L	6.18 (3.90)	4.73 (3.34)	5.67 (5.94)	4.37 (4.03)	5.24 (4.30)
		XL	6.03 (3.38)	4.80 (3.13)	5.55 (5.52)	4.24 (3.91)	5.15 (3.98)
Demucs v3-B [19]	200+ including *musdb-hq* test set	*hq*	**9.24 (7.05)**	**11.65 (9.58)**	**11.73 (11.34)**	**7.83 (8.03)**	**10.11 (9.00)**
		L	9.05 (6.91)	11.61 (9.55)	11.05 (10.27)	7.83 (7.91)	9.88 (8.66)
		XL	8.76 (6.41)	11.56 (9.29)	9.22 (8.78)	7.52 (7.51)	9.26 (8.00)

Table 2. Performance of various music source separation networks trained with *musdb-hq* [14] on the test using *musdb-hq*, *musdb-L* and *musdb-XL*.

In Section 5.2, we simply normalized the network inputs in the evaluation stage, to observe the performance degradation caused by the signal distortion caused by a limiter, not an overall amplitude mismatch between the training and the evaluation data. Then, we carefully analyzed the results on *Demucs v3* [19], which took the 1st place in the 2021 Sony Music Demixing (MDX) Challenge leaderboard A and the 2nd place in the leaderboard B [22]. The leaderboard A was the competition that only allowed training on the only *musdb-hq* train subset, and the leaderboard B allowed the extra training data.

In Section 5.3, we conducted a comparative study on various training data construction strategies. Unfortunately, we were unable to train the current state-of-the-art *Demucs v3*, due to the constraint of our GPU experimental environments. Considering the training efficiency and reasonable performance, we used *TFC-TDF-U-Net* [20] — a backbone architecture of KUIELab-MDX-Net [23] with slight modifications, which took the 2nd place in the 2021 MDX Challenge leaderboard A — in this experiment.

4.1 Training

In Section 5.1 and Table 2, we used official pre-trained weights of each model except *TFC-TDF-U-Net* since there were no official weights for the 4 stems of *musdb-hq*. For fair comparison in Section 5.3, we trained *TFC-TDF-U-Net* for 300 epochs with early stopping, based on official training framework of *Open-unmix* [6] and official network implementation of *TFC-TDF-U-net*. We used default network hyper-parameters introduced in its webpage [5] .

For the *LimitAug*, we used the limiter implemented in `pedalboard` [24]. The threshold parameter was set to 0 dBFS, and the release parameter was randomly sampled from uniform distribution of (30, 200) millisecond in data sampling process. Also, we used `pyloudnorm` [25] for loudness calculation.

[5] https://github.com/ws-choi/ISMIR2020_U_Nets_SVS

4.2 Evaluation

In all of the evaluations, Signal-to-Distortion Ratio (SDR) [26] was calculated using `museval` python library [18]. Also, in following experimental results, both *median* and *mean* SDR scores were presented for the detailed comparison. Note that we only used the *musdb-hq* train subset for training the networks. *musdb-L* and *XL* are only for evaluation.

In the original *musdb-hq* test subset, as stated in the official webpage [6] , 'PR - Oh No' track's mixture signal is panned to the right channel, which causes the inconsistency between linear summation of stems and mixture source. Since this causes the limiter to be operated in unmusical way while the construction of *musdb-L* and *XL*, which also can be hardly found in popular music, we used the linear summation of stems as a mixture only for this track. This may results in the slight difference of SDR scores between the Table 2 and the official scores of each network.

5. RESULTS

5.1 Performance degradation on *musdb-L* and *XL*

Here we quantitatively evaluated the performance of state-of-the-art networks on *musdb-hq* [14], *L* and *XL* datasets and confirmed that the domain shift in perspective of music loudness and compression negatively affect the performance, indeed. As shown in the Table 2, all networks showed significant performance degradation on the evaluations with *musdb-L* and *musdb-XL* datasets. The amount of decrease on *Demucs v3* [19] was slightly larger than the others. Overall, we concluded that every networks are somewhat overfitted to the *musdb-hq* data, making the networks vulnerable to loud and heavily compressed music. Therefore, it is highly required to consider these characteristics for the robust music source separation. Note that

[6] https://sigsep.github.io/datasets/musdb

network	extra train data	test data	SDR median (mean) [dB]				
			vocals	*bass*	*drums*	*other*	*avg*
Demucs v3-A [19]	-	*hq*	**8.11** (5.22)	**9.34 (6.21)**	**8.57 (8.01)**	5.51 (**5.03**)	**7.88 (6.12)**
		L	8.05 (**5.23**)	9.25 (6.20)	8.47 (7.92)	**5.53** (5.02)	7.82 (6.09)
		XL	7.93 (5.03)	9.27 (5.92)	7.74 (7.44)	5.55 (4.91)	7.62 (5.82)
Demucs v3-B [19]	200+ including *musdb-hq* test set	*hq*	**9.24 (7.05)**	**11.65 (9.58)**	**11.73 (11.34)**	**7.83 (8.03)**	**10.11 (9.00)**
		L	9.19 (7.04)	11.64 (9.55)	11.68 (11.21)	7.82 (8.02)	10.08 (8.95)
		XL	9.13 (6.90)	11.56 (9.33)	11.32 (10.75)	7.74 (7.95)	9.94 (8.73)

Table 3. Performance of *Demucs v3* [19] trained with *musdb-hq* [14]. On the test using *musdb-L* and *musdb-XL*, each of the input sources were loudness normalized targeting the LUFS of its original *musdb-hq* sources.

network	extra train data	SDR median [dB]		
		hq	*L*	*XL*
Open-unmix [6]	-	5.35	5.32	5.25
TFC-TDF-U-Net [20]	-	5.64	5.62	5.51
Demucs v3-A [19]	-	**7.88**	**7.82**	**7.62**
Open-unmix [6]	✓	6.26	6.25	6.18
Spleeter [21]	✓	5.38	5.33	5.21
Demucs v3-B [19]	✓	**10.11**	**10.08**	**9.94**

Table 4. Performance of networks with loudness normalized input at the evaluation stage. Each score represents the average score across the 4 stems. Note that the networks were trained without considering music loudness or dynamic range compression explicitly.

we stated the scores on different test datasets in each block to emphasize the performance degradation between the domains.

5.1.1 Extra training data

Though the extra training data was of help, it did not work as the fundamental solution to the domain shift. *Open-unmix* [6] with extra training data showed more robust performance than that of the network without extra data, but performance degradation on *Demucs v3-B* was more significant than that of *Demucs v3-A*. Since *Demucs v3-B* was trained with extra training data including *musdb-hq* test subset, it is reasonable to guess that the network is highly overfitted to the *musdb-hq* data. Nevertheless, the amount of performance decrease especially on *drums* comparing the scores between *musdb-hq* and *XL*, 2.5 dB, is somewhat high. Considering *Demucs v3* uses an input standardization technique based on mean and standard deviation of waveform values both at training and evaluation stages, this implies that there needs to be another solution for the robust music source separation.

5.1.2 Performance degradation on drums and vocals

It is noteworthy that the degradation on *drums* and *vocals* were more significant than the others. Due to the percussive nature of *drums*, in general, they have the biggest momentary energy in music. Also, since *vocals* are important ingredients in modern commercial music, they usually consist of not only a single singing voice but also doubling, harmony and chorus. Therefore, we assumed *drums* and *vocals* are most affected by the limiter, which is activated when loud input sources that are above the threshold are given. To quantitatively confirm the signal distortion by the limiter, we calculated Scale-invariant Signal-to-Distortion Ratio (SI-SDR) [27] between *musdb-hq* and *musdb-XL* for each stem. As a result, *drums* and *vocals* scored each 19.97 and 23.69 dB, on the other hand, *bass* and *other* scored each 25.12 and 25.48 dB on average. That is, the signal distortion caused by a limiter is more significant on *drums* and *vocals*.

Unfortunately, since the networks in the Table 2 have never seen these kinds of distorted *drums* or *vocals* as training examples, we assumed that the degradation on *drums* and *vocals* were significant compared to the rest. This result strongly supports the necessity of considering the heavy dynamic range compression from the training stage, i.e. the *LimitAug*, which forcibly makes the distorted and compressed training examples for training purposes, thereby minimizing the domain shift.

5.2 Analysis on the input normalization at the evaluation stage

If the key differences between the real-world music and the training examples of music source separation networks are *i)* overall amplitude scale, and *ii)* the signal distortion caused by a limiter, as stated in Section 3, then what if we give the loudness normalized *musdb-L* or *XL* data as inputs to the networks in Table 2? Due to the non-linear nature of deep neural networks, we assumed that simply normalizing the amplitude scale of the networks on the evaluation stage may affect the performance.

The inference was conducted with following procedures, *(i)* reducing the loudness of *musdb-L* or *XL* input so that its loudness becomes same with that of the corresponding original *musdb-hq* data, *(ii)* inference with loudness normalized input, and *(iii)* increasing the scale of output as much as reduced in step *(i)*.

Comparing the scores between Table 2 and Table 4, we confirmed that the performance degradation was greatly alleviated by just the simple loudness normalization of the inputs only at the evaluation stage. Note that the networks were not trained with loudness normalized inputs. This result shows that the input loudness normalization at the evaluation stage can be a quick and easy solution to get robust results from the pre-trained music source separation networks.

Especially, on *drums* of *Demucs v3-B*, it should be noted that the amount of decrease on *median* SDR score between the test using *musdb-hq* and *XL* was sharply reduced from 2.5 dB in Table 2 to 0.4 dB in Table 3. This implies that the network is overfitted not only to the contents of the signal, but also to the scale or loudness, especially

network	methods	linear gain increase	*LimitAug*	input loud-norm	target LUFS	SDR median (mean) [dB]			
						hq	*L*	*XL*	*avg*
TFC-TDF -U-Net [20]	baseline	-	-	-	-	5.64 (4.02)	5.41 (3.71)	5.27 (3.39)	5.44 (3.71)
	(1)	✓	-	-	$\mathcal{N}(\mu_L, \sigma_L^2)$	**5.90** (4.31)	5.86 (4.33)	5.73 (4.15)	5.83 (4.26)
					$\mathcal{N}(\mu_{XL}, \sigma_{XL}^2)$	5.32 (3.43)	5.36 (3.62)	5.28 (3.49)	5.32 (3.51)
	(2)	-	✓	-	$\mathcal{N}(\mu_L, \sigma_L^2)$	5.79 (4.30)	**5.90 (4.41)**	5.74 (4.25)	5.81 (4.32)
					$\mathcal{N}(\mu_{XL}, \sigma_{XL}^2)$	5.69 (3.93)	5.72 (4.22)	5.57 (4.15)	5.66 (4.10)
	(3)	-	-	✓	-14	5.89 **(4.38)**	5.87 (4.35)	**5.82 (4.25)**	**5.86 (4.33)**
	(4)	-	✓	✓	$\mathcal{N}(\mu_L, \sigma_L^2)$, -14	5.87 (4.25)	5.85 (4.21)	5.76 (4.16)	5.83 (4.21)
					$\mathcal{N}(\mu_{XL}, \sigma_{XL}^2)$, -14	5.78 (4.27)	5.78 (4.26)	5.73 (4.20)	5.76 (4.24)

Table 5. Performance of *TFC-TDF-U-Net* [20] trained with various training data construction strategies. Each score represents the average score across the 4 stems.

network	methods	target LUFS	test data	SDR median (mean) [dB]				
				vocals	*bass*	*drums*	*other*	*avg*
TFC-TDF -U-Net [20]	baseline	-	*hq*	7.18 (4.26)	5.59 (3.35)	5.76 (5.30)	4.04 (3.18)	5.64 (4.02)
			L	7.03 (3.65)	5.41 (3.08)	5.52 (5.09)	3.67 (3.00)	5.41 (3.71)
			XL	6.95 (3.14)	5.48 (2.90)	5.11 (4.68)	3.55 (2.82)	5.27 (3.39)
	(3) loud-norm	-14	*hq*	7.35 **(4.76)**	**5.93** (3.61)	**5.91 (5.37)**	4.39 (3.79)	**5.89 (4.38)**
			L	7.32 (4.72)	5.91 (3.61)	5.85 (5.29)	4.39 (3.78)	5.87 (4.35)
			XL	7.26 (4.64)	5.91 **(3.62)**	5.68 (4.99)	4.42 (3.78)	5.82 (4.25)
	(4) LimitAug, loud-norm	$\mathcal{N}(\mu_L, \sigma_L^2)$, -14	*hq*	**7.59** (4.64)	5.75 (3.25)	5.63 (5.28)	4.50 (3.82)	5.87 (4.25)
			L	7.58 (4.61)	5.69 (3.21)	5.62 (5.22)	4.50 (3.82)	5.85 (4.21)
			XL	7.48 (4.55)	5.67 (3.29)	5.36 (4.99)	**4.51 (3.82)**	5.76 (4.16)

Table 6. Stem-wise performance of *TFC-TDF-U-Net* [20] trained with the method *(3)* input loudness normalization, and *(4)* input loudness normalization after the proposed *LimitAug*.

on *drums*. Note that there was no distinction between the given input sources to the networks except the linear gain difference.

Although the performance degradation was reduced by the input loudness normalization, still there exists the performance degradation due to the signal distortion caused by a limiter. Similar to Section 5.1.2, this result also supports the necessity of the *LimitAug* for robust music source separation.

5.3 Analysis on various training strategies

In this section, we trained the *TFC-TDF-U-net* [20] with various training data creating methods; *(1)* linear gain increasing, *(2)* the proposed *LimitAug*, *(3)* input loudness normalization, and *(4)* input loudness normalization after the *LimitAug*. Of course, the methods *(3)* and *(4)* includes the input loudness normalization at the evaluation stage for the consistency between train and test domains. We compared the results to check which one is the most powerful way for training robust music source separation networks. For the input normalization, we chose the target reference LUFS value as -14.

In Table 5, we confirmed that all of the methods were effective for robust music source separation; every methods showed relatively robust performance on *musdb-L* and *XL*, compared to the baseline. Especially, except the method *(1)* targeting LUFS of a normal distribution following statistics of *musdb-XL*, $\mathcal{N}(\mu_{XL}, \sigma_{XL}^2)$, all methods showed greater performance on *musdb-hq* than the baseline. This result implies that these methods are useful not only for the domain shift, but also for the standard benchmark data.

Furthermore, the methods *(3)* and *(4)*, which prevent the domain shift caused by overall amplitude scale mismatch by input loudness normalization, showed slightly better performances than others. In the stem-wise analysis as presented in Table 6, though we expected that the *LimitAug* would be of help for *vocals* and *drums*, the method *(4)* was better at *vocals* and *other* than the method *(3)*.

Overall, it seems obvious that considering the music loudness and heavy dynamic range compression from the training stage is beneficial for robust music source separation. For real world applications, it is highly recommended that not just using the single method we experimented, but using various training methods on different stems or using a bag of models trained with various methods.

6. CONCLUSIONS

In this study, we questioned the domain shift between the research and the real-world data for music source separation, from the viewpoint of music loudness and heavy dynamic range compression. To answer this, We first built new evaluation datasets, *musdb-L* and *musdb-XL*, which reflect dynamic range compressed music characteristics and heavy loudness. Then, we confirmed the significant performance degradation of state-of-the-art networks on our datasets. To alleviate this, we conducted various experiments on training data construction strategies, including the proposed *LimitAug* method, and confirmed that the methods using the input loudness normalization only or with the *LimitAug* greatly improved the robustness. We hope that our proposed methods and evaluation datasets could contribute to future music source separation research and application.

7. ACKNOWLEDGEMENTS

We appreciate Zafar Rafii, the author of *musdb*, for allowing us to reprocess the original musdb data. We thank Antoine Liutkus, also the author of *musdb*, for giving the creative suggestion on the distribution of our proposed datasets. We are grateful to Ben Sangbae Chon, Keunwoo Choi, and Hyeongi Moon from GaudioLab for their fruitful discussions on our proposed methods. Last but not least, we thank Donmoon Lee, Juheon Lee, Jaejun Lee, Junghyun Koo, and Sungho Lee for helpful feedbacks.

8. REFERENCES

[1] S. Dredge, "Pop music is louder, less acoustic and more energetic than in the 1950s," *The Guardian*, 25 Nov. 2013.

[2] G. Milner, "They really don't make music like they used to," *The New York Times*, 7 Feb. 2019.

[3] N. B. Croghan, K. H. Arehart, and J. M. Kates, "Quality and loudness judgments for music subjected to compression limiting," *The Journal of the Acoustical Society of America*, vol. 132, no. 2, pp. 1177–1188, 2012.

[4] E. Vickers, "The loudness war: Background, speculation, and recommendations," in *Audio Engineering Society Convention 129*. Audio Engineering Society, 2010.

[5] E. F. Stikvoort, "Digital dynamic range compressor for audio," *Journal of the Audio Engineering Society*, vol. 34, no. 1/2, pp. 3–9, 1986.

[6] F.-R. Stöter, S. Uhlich, A. Liutkus, and Y. Mitsufuji, "Open-unmix - a reference implementation for music source separation," *Journal of Open Source Software*, 2019. [Online]. Available: https://doi.org/10.21105/joss.01667

[7] B. Katz and R. A. Katz, *Mastering audio: the art and the science*. Butterworth-Heinemann, 2003.

[8] L. Bregitzer, *Secrets of recording: Professional tips, tools & techniques*. Routledge, 2008.

[9] S. Uhlich, M. Porcu, F. Giron, M. Enenkl, T. Kemp, N. Takahashi, and Y. Mitsufuji, "Improving music source separation based on deep neural networks through data augmentation and network blending," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 261–265.

[10] L. Prétet, R. Hennequin, J. Royo-Letelier, and A. Vaglio, "Singing voice separation: A study on training data," in *ICASSP 2019-2019 ieee international conference on acoustics, speech and signal processing (icassp)*. IEEE, 2019, pp. 506–510.

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

[12] R. EBU-Recommendation, "Loudness normalisation and permitted maximum level of audio signals," *European Broadcasting Union*, 2011.

[13] 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

[14] Z. Rafii, A. Liutkus, F.-R. Stöter, S. I. Mimilakis, and R. Bittner, "Musdb18-hq - an uncompressed version of musdb18," Aug. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3338373

[15] E. Manilow, G. Wichern, P. Seetharaman, and J. Le Roux, "Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity," in *2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*. IEEE, 2019, pp. 45–49.

[16] A. Liutkus, F.-R. Stöter, Z. Rafii, D. Kitamura, B. Rivet, N. Ito, N. Ono, and J. Fontecave, "The 2016 signal separation evaluation campaign," in *International conference on latent variable analysis and signal separation*. Springer, 2017, pp. 323–332.

[17] R. M. Bittner, J. Salamon, M. Tierney, M. Mauch, C. Cannam, and J. P. Bello, "Medleydb: A multitrack dataset for annotation-intensive mir research." in *ISMIR*, vol. 14, 2014, pp. 155–160.

[18] F.-R. Stöter, A. Liutkus, and N. Ito, "The 2018 signal separation evaluation campaign," in *Latent Variable Analysis and Signal Separation: 14th International Conference, LVA/ICA 2018, Surrey, UK*, 2018, pp. 293–305.

[19] A. Défossez, "Hybrid spectrogram and waveform source separation," *arXiv preprint arXiv:2111.03600*, 2021.

[20] W. Choi, M. Kim, J. Chung, D. Lee, and S. Jung, "Investigating u-nets with various intermediate blocks for spectrogram-based singing voice separation," in *21th International Society for Music Information Retrieval Conference, ISMIR, Ed*, 2020.

[21] R. Hennequin, A. Khlif, F. Voituret, and M. Moussallam, "Spleeter: a fast and efficient music source separation tool with pre-trained models," *Journal of Open Source Software*, vol. 5, no. 50, p. 2154, 2020.

[22] Y. Mitsufuji, G. Fabbro, S. Uhlich, and F.-R. Stöter, "Music demixing challenge 2021," *arXiv preprint arXiv:2108.13559*, 2021.

[23] M. Kim, W. Choi, J. Chung, D. Lee, and S. Jung, "Kuielab-mdx-net: A two-stream neural network for music demixing," *arXiv preprint arXiv:2111.12203*, 2021.

[24] Spotify, "pedalboard : A python library for manipulating audio." 2022. [Online]. Available: https://github.com/spotify/pedalboard

[25] C. J. Steinmetz and J. D. Reiss, "pyloudnorm: A simple yet flexible loudness meter in python," in *150th AES Convention*, 2021.

[26] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE transactions on audio, speech, and language processing*, vol. 14, no. 4, pp. 1462–1469, 2006.

[27] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "Sdr–half-baked or well done?" in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 626–630.