ON LOSS FUNCTIONS AND EVALUATION METRICS FOR MUSIC SOURCE SEPARATION

Enric Gusó[⋆], Jordi Pons[†], Santiago Pascual[†], Joan Serrà[†]

Music Technology Group, Universitat Pompeu Fabra[⋆]   Dolby Laboratories[†]

ABSTRACT

We investigate which loss functions provide better separations via benchmarking an extensive set of those for music source separation. To that end, we first survey the most representative audio source separation losses we identified, to later consistently benchmark them in a controlled experimental setup. We also explore using such losses as evaluation metrics, via cross-correlating them with the results of a subjective test. Based on the observation that the standard signal-to-distortion ratio metric can be misleading in some scenarios, we study alternative evaluation metrics based on the considered losses.

Index Terms— source separation, loss functions, evaluation.

1. INTRODUCTION

Sound mixing consists in bringing together different audio signals (sources) to create a mix. Source separation is the inverse process of un-mixing, with the goal of recovering the original signals. Our work focuses on music source separation, and on which training objectives (losses) provide separations that are perceptually preferable. To that end, we benchmark an extensive set of regression losses that are representative of the recent literature on audio source separation (Fig. 1). Regression losses are also often used as evaluation metrics, since they are a quick alternative to subjective tests [1, 2, 3]. However, these can be problematic [4]. Thus, we also study their appropriateness for evaluation by correlating them with human judgment.

Many successful music source separation methods rely on regressors operating on the time-frequency domain. These can be used to estimate the sources' magnitude spectrograms [5, 6], or to estimate separation masks to filter the magnitude spectrogram of the mixture. While spectrogram-based models are widely used [5, 6, 7], these can be problematic as they often discard the phase, which is needed to go back to the time domain. To solve that, the phase of the mixture is often used [5, 6, 7], but this can introduce errors. Our work considers the use of phase-sensitive and time-domain losses to capture phase errors within the training objective. Further, note that scale-invariant time-domain losses are now becoming popular for audio source separation [1, 3]. Our work also investigates the use of scale-invariant regression losses for spectrogram signals [8]. Finally, if we look at training losses from a machine learning perspective, it is known that minimizing the $L1$ or $L2$ loss leads to predicting the median or the mean [9], respectively, of the output distribution that is assumed to be uni-modal. Consequently, if the underlying distribution is not as assumed, predictions may not follow the true distribution, introducing errors. This is commonly tackled with regularizers: adversarial losses that help fitting a "realistic" mode, and deep feature losses that provide additional cues during training.

Our main contributions are: *i)* a survey of the most representative losses for neural audio source separation, in section 2; *ii)* to benchmark such losses, in sections 3 and 4; and *iii)* a subjective evaluation to study which losses correlate best with human judgments, in section 5. Note that to consistently benchmark the considered losses one needs to choose a specific model. We select OpenUnmix [5] because it is an open-source[1] state-of-the-art model based on predicting time-frequency masks, what is significantly faster to train than waveform-based models [10, 11].

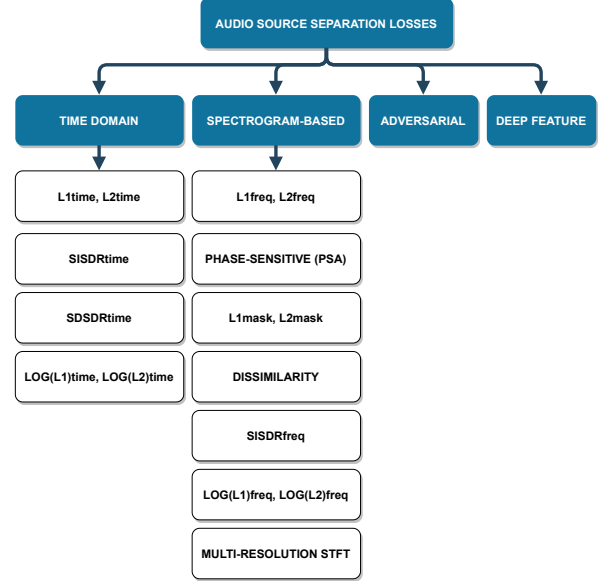[1] https://github.com/sigsep/open-unmix-pytorch

Fig. 1. Taxonomy of the considered audio source separation losses.

2. AUDIO SOURCE SEPARATION LOSSES

2.1. Time domain losses

Given an audio x_t of $\mathcal{T}$ samples, we extract K different sources $\tilde{y}_{t,k}$. Standard regression losses, like L1 or L2, have been used to train audio source separation models [5, 6]:

$$\text{L1}_{\text{time}} = \frac{1}{\mathcal{T}K} \sum_{t,k} |\tilde{y}_{t,k} - y_{t,k}|,$$

$$\text{L2}_{\text{time}} = \frac{1}{\mathcal{T}K} \sum_{t,k} |\tilde{y}_{t,k} - y_{t,k}|^2.$$

Further, signal-to-distortion rartio (SDR) [12] and scale-invariant SDR (SISDR) [1] are widely used for evaluation and training [13]:

$$\text{SISDR}_{\text{time}} = \frac{-10}{\mathcal{T}K} \sum_{t,k} \log_{10} \left(\frac{\left| \frac{\tilde{y}_{t,k}^{\text{T}} y_{t,k}}{|y_{t,k}|^2} y_{t,k} \right|^2}{\left| \frac{\tilde{y}_{t,k}^{T} y_{t,k}}{|y_{t,k}|^2} y_{t,k} - \tilde{y}_{t,k} \right|^2} \right).$$

However, in some cases, the scale-invariance assumption might not hold and the scale-dependent SDR (SDSDR) [1] is preferable:

$$\text{SNR} = \frac{10}{\mathcal{T}K} \sum_{t,k} \log_{10} \left(\frac{|y_{t,k}|^2}{|\tilde{y}_{t,k} - y_{t,k}|^2} \right),$$

$$\text{L}_{\text{down}} = SNR + \frac{10}{\mathcal{T}K} \sum_{t,k} \log_{10} \left(\left(\frac{\tilde{y}_{t,k}^{\text{T}} y_{t,k}}{|y_{t,k}|^2} \right)^2 \right),$$

$$\text{SDSDR}_{\text{time}} = -\min(\text{SNR}, \text{L}_{\text{down}}).$$

ICASSP 2022

Further, Heitkaemper et al. [14] derived a log-compressed L2 loss, equivalent to SISDR. We also experiment with an L1 variant:

$$\text{LOGL2}_{\text{time}} = \frac{10}{\mathcal{T}K} \sum_k \log_{10} \sum_t |\tilde{y}_{t,k} - y_{t,k}|^2,$$

$$\text{LOGL1}_{\text{time}} = \frac{10}{\mathcal{T}K} \sum_k \log_{10} \sum_t |\tilde{y}_{t,k} - y_{t,k}|.$$

2.2. Spectrogram-based losses

We define the short-time Fourier transform as $STFT(x_t) = X_{n,\omega}$, where n and ω stand for frame and frequency bin indices, respectively, of a total number of N frames and Ω frequency bins. Many source separation models estimate a magnitude spectrogram $|\tilde{Y}_{n,\omega,k}|$ per source k, and rely on standard regression losses like L1 or L2:

$$\text{L1}_{\text{freq}} = \frac{1}{N\Omega K} \sum_{n,\omega,k} \left| \, |\tilde{Y}_{n,\omega,k}| - |Y_{n,\omega,k}| \, \right|,$$

$$\text{L2}_{\text{freq}} = \frac{1}{N\Omega K} \sum_{n,\omega,k} \left| \, |\tilde{Y}_{n,\omega,k}| - |Y_{n,\omega,k}| \, \right|^2.$$

Note that $|\tilde{Y}_{n,\omega,k}|$ does not include phase information. To synthesize the time-domain signal from $|\tilde{Y}_{n,\omega,k}|$ with the inverse short-time Fourier transform, phase information is required. Often times, the noisy phase of the input mixture $\angle X_{n,\omega}$ is used: $\tilde{y}_{t,k} = ISTFT(|\tilde{Y}_{n,\omega,k}|\angle X_{n,\omega})$, which can introduce errors. To mitigate those, a phase sensitive target $Y_{n,\omega,k}^{\text{PSA}}$ was proposed [15]:

$$Y_{n,\omega,k}^{\text{PSA}} = |Y_{n,\omega,k}| \cos(\angle X_{n,\omega} - \angle Y_{n,\omega,k}),$$

$$\text{L}_{\text{PSA}} = \frac{1}{N\Omega K} \sum_{n,\omega,k} \left| \, |\tilde{Y}_{n,\omega,k}| - Y_{n,\omega,k}^{\text{PSA}} \, \right|^2.$$

Further, mask-based models are commonly used when operating with spectrogram signals and, e.g., ratio masks are widely employed [5, 16] to filter the sources out from the mixture $X_{n,\omega}$:

$$M_{n,\omega,k} = \frac{|Y_{n,\omega,k}|}{\sum_k |Y_{n,\omega,k}|},$$

$$|Y_{n,\omega,k}| = M_{n,\omega,k} \odot |X_{n,\omega}|.$$

Standard regression losses are used to train mask-based models [5]:

$$\text{L1}_{\text{mask}} = \frac{1}{N\Omega K} \sum_{n,\omega,k} |\tilde{M}_{n,\omega,k} - M_{n,\omega,k}|,$$

$$\text{L2}_{\text{mask}} = \frac{1}{N\Omega K} \sum_{n,\omega,k} |\tilde{M}_{n,\omega,k} - M_{n,\omega,k}|^2.$$

The above losses encourage separations to be similar to the targets. Yet, source separation models are prone to introduce interferences from other sources. To mitigate this, Huang et al. [17] proposed a dissimilarity loss encouraging each prediction to be different from the rest by adding an L2 term for each additional source [17, 6]:

$$\text{L}_{\text{dissim}} = \frac{1}{N\Omega K} \sum_{n,\omega,k} \left(\left| \, |\tilde{Y}_{n,\omega,k}| - |Y_{n,\omega,k}| \, \right|^2 - \right.$$

$$\left. -\beta \sum_{\tilde{k}} \left| \, |\tilde{Y}_{n,\omega,k}| - |Y_{n,\omega,\tilde{k}\neq k}| \, \right|^2 \right).$$

We also adapt SISDR$_{\text{time}}$ to work with spectrograms, reshaping $Y_{n,k,\omega}$ and $\tilde{Y}_{n,k,\omega}$ into $Y_{n\omega,k}$ and $\tilde{Y}_{n\omega,k}$, respectively. Further, we adapt LOGL1$_{\text{time}}$ and LOGL2$_{\text{time}}$ for spectrograms:

$$\text{LOGL1}_{\text{freq}} = \frac{10}{N\Omega K} \sum_k \log_{10} \sum_{n,\omega} \left| \, |\tilde{Y}_{n,\omega,k}| - |Y_{n,\omega,k}| \, \right|,$$

$$\text{LOGL2}_{\text{freq}} = \frac{10}{N\Omega K} \sum_k \log_{10} \sum_{n,\omega} \left| \, |\tilde{Y}_{n,\omega,k}| - |Y_{n,\omega,k}| \, \right|^2.$$

We also study the multi-resolution STFT loss (L$_{\text{MRS}}$), implemented via L$_{\text{sc}}$ (spectral convergence) and L$_{\text{mag}}$ (log magnitude) [18, 19]. It is composed of single-resolution STFTs with different configurations O having various FFT sizes, windows and overlaps:

$$\text{L}_{\text{sc}} = \frac{1}{K} \sum_k \frac{\sqrt{\sum_{n,\omega} (\, |\tilde{Y}_{n,\omega,k}| - |Y_{n,\omega,k}| \,)^2}}{\sqrt{\sum_{n,\omega} |Y_{n,\omega,k}|^2}},$$

$$\text{L}_{\text{mag}} = \frac{1}{K} \sum_{n,\omega,k} (\log_{10} |\tilde{Y}_{n,\omega,k}| - \log_{10} |Y_{n,\omega,k}|),$$

$$\text{L}_{\text{MRS}} = \frac{1}{O} \sum_o \left(\text{L}_{\text{sc}}^{(o)} + \text{L}_{\text{mag}}^{(o)} \right).$$

2.3. Adversarial loss

Our adversarial loss [20, 21, 22] (Fig. 2) consists of *i)* a separator f producing plausible separations $\tilde{Y}$ (magnitude spectrograms) from paired X and unpaired $\hat{X}$ mixtures; and *ii)* several discriminators D, dictating whether they are produced by the separator $\tilde{Y}$ (fake) or come from a database of paired Y or unpaired $\hat{Y}$ isolated recordings (real). As a result, this setup allows using additional (unpaired) data to train the separator (see in Fig. 2). We use a least-squares variant [23] with label smoothing [24, 25], and pre-train the separator with an L2 loss (until convergence) to help with training stability. We train k DCGAN-like discriminators [26, 22], one per source:

$$\text{L}_k^{\text{real\_unpair}} = \frac{1}{4} \left(D_k(\hat{Y}_{n,\omega,k}) - 0.9 \right)^2,$$

$$\text{L}_k^{\text{real\_pair}} = \frac{1}{4} \left(D_k(Y_{n,\omega,k}) - 0.9 \right)^2,$$

$$\text{L}_k^{\text{fake\_unpair}} = \frac{1}{4} \left(D_k(f(\hat{X}_{n,\omega})_k) - 0.1 \right)^2,$$

$$\text{L}_k^{\text{fake\_pair}} = \frac{1}{4} \left(D_k(f(X_{n,\omega})_k) - 0.1 \right)^2,$$

$$\text{L}_k = \text{L}_k^{\text{real\_unpair}} + \text{L}_k^{\text{real\_pair}} + \text{L}_k^{\text{fake\_unpair}} + \text{L}_k^{\text{fake\_pair}}.$$

With each iteration, we first update each discriminator k (without updating the separator) based on the L$_k$ loss. Then, the L$_{\text{adv}}$ loss is used to train the separator (without updating the discriminator):

$$\text{L}_{\text{sep}} = \frac{1}{K} \sum_k (D_k(f(X_{n,\omega})_k) - 0.9)^2 + (D_k(f(\hat{X}_{n,\omega})_k) - 0.9)^2,$$

$$\text{L}_{\text{adv}} = \text{L2}_{\text{freq}} + \gamma \text{L}_{\text{sep}}.$$

2.4. Deep feature loss

Given a neural network ϕ with J layers, the target Y and estimated $\tilde{Y}$ magnitude spectrograms can be projected through ϕ to obtain their embeddings. Provided that ϕ is differentiable, the distance between target and estimated audio embeddings can be used to define deep
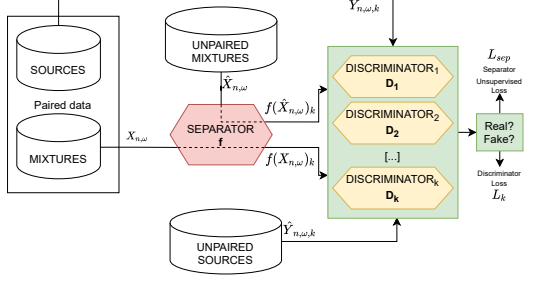
Fig. 2. Adversarial loss schema.

feature losses at every layer j of ϕ [27]. The deep feature loss we use is based on the L2 distance [28]:

$$\mathrm{L}_{\mathrm{feat}}^{\phi,j} = \frac{1}{C_j H_j W_j K} \sum_{c,h,w,k} |\phi_j(\tilde{Y}_{n,\omega,k}) - \phi_j(Y_{n,\omega,k})|^2,$$

where $C_j \times H_j \times W_j$ is the embedding's shape at each layer j. Deep feature losses were extended with a style reconstruction loss $\mathrm{L}_{\mathrm{sty}}^{\phi,j}$ capturing information on which features activate together [27, 28]. To that end, first, the gram matrix $G_j^{\phi(Y)}$ is defined as

$$G_j^{\phi(Y)} = \frac{\psi_Y \psi_Y^{\mathrm{T}}}{C_j H_j W_j},$$

where ψ_Y corresponds to Y's embeddings reshaped as $C_j \times H_j Wj$. Accordingly, $G_j^{\phi(Y)}$ is of shape $C_j \times C_j$. Then:

$$\mathrm{L}_{\mathrm{sty}}^{\phi,j} = \sum_{c,c} \left| G_j^{\phi(\tilde{Y})} - G_j^{\phi(Y)} \right|^2.$$

Our ϕ is a VGGish [29] model pre-trained on AudioSet [30], and we use the above deep feature losses to regularize an $\mathrm{L2}_{\mathrm{freq}}$ loss [28]:

$$\mathrm{L}_{\mathrm{deep\_feat}} = \mathrm{L2}_{\mathrm{freq}} + \delta \mathrm{L}_{\mathrm{feat}}^{\phi,j} + \lambda \mathrm{L}_{\mathrm{sty}}^{\phi,j}$$

3. EXPERIMENTAL SETUP

Task — We approach music source separation as defined by the MUSDB18 dataset [31], consisting of recovering 4 stereo stems (drums, bass, vocals and the residual 'other') from stereo mixes. MUSDB18 includes 150 songs (86 train, 14 val, 50 test) at 44.1kHz.

Model — As noted above, we rely on OpenUnmix [5] to consistently benchmark different training losses. Its original form is trained to independently separate vocals, bass, drums & 'other'. Hence, it trains a model per source. OpenUnmix later joins all these predictions via a multichannel Wiener post-filter while ensuring that all separations sum up to the mixture, further improving the separation [5]. To simplify the pipeline, we modify OpenUnmix to predict all four sources using a single joint model: jOpenUnmix (Fig. 3). In addition, we increase its latent from 512 feature maps to 1024. These minor modifications improve the results (trained with the original $\mathrm{L2}_{\mathrm{freq}}$ loss): vocals from 6.32 dB to 6.4 dB, drums from 5.73 dB to 5.86 dB, bass from 5.23 dB to 5.28 dB, and 'other' from 4.02 dB to 4.57 dB. jOpenUnmix keeps the original multichannel Wiener post-filtering.

Losses setup — For the adversarial loss, we set $\gamma = 0.5$ [22] and use an additional dataset where mixtures and isolated track recordings are unpaired. We use 2065 additional unpaired mixtures from the Million Song Dataset [32]. The unpaired vocals are 258 tracks
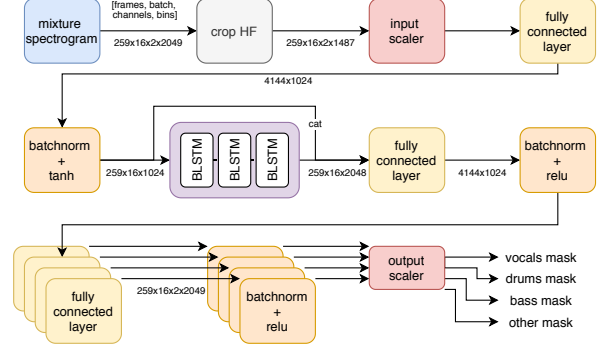

Fig. 3. jOpenUnmix model, the baseline.

from the iKala dataset [33] and 58 vocals tracks from the FreeSound Dataset (FSD) [34]. The unpaired drums and bass database consists of 114 and 83 tracks also from FSD, respectively. The 99 'other' tracks are from FSD—with the tags keys, brass, guitar, and strings. Our DCGAN-like discriminators operate over stereo spectrograms and each consists of 9 CNN layers (of 4×4 and stride of 2×2 with 16, 32, 64, 128, 256, 512, 512, 512, 512 channels) and an MLP (with 3 layers of 1024, 32, 1 channels). We set $\beta = 0.05$ for $\mathrm{L}_{\mathrm{dissim}}$ [17]. We set $\delta = \frac{1}{2}$ and $\lambda = \frac{10}{3}$ for $\mathrm{L}_{\mathrm{deep\_feat}}$ [28]. $\mathrm{L}_{\mathrm{MRS}}$ uses windows of 2048 and 1024 with 512 and 256 hop sizes. We use differentiable inverse-STFT layers for the losses requiring access to waveforms.

Learning rates — Models are trained until convergence with Adam. We reduce the learning rate $\times 0.3$ after the validation loss does not improve for 80 epochs. It was key to independently adjust the learning rate for each loss. Generally, 10^{-3} performed the best—except for 10^{-4} with $\mathrm{L2}_{\mathrm{mask}}$ loss, and 10^{-5} with $\mathrm{L1}_{\mathrm{time}}$ and $\mathrm{L2}_{\mathrm{time}}$ losses. Our discriminators and separator also use a learning rate of 10^{-3}.

4. OBJECTIVE EVALUATION

We employ two sets of evaluation metrics: *i)* the commonly used signal-to-distortion ratio (SDR, the higher the better) [12, 35], in Table 1; and *ii)* the losses above used as evaluation metrics (the lower the better), in Fig. 4. In order to visually navigate the loss-based metrics in Fig. 4, we standardize them to be in the same range.[2] As a result, columns intuitively convey the performance of each loss across a wide range of loss-based metrics (vertical red lines denote bad performance, and blueish vertical lines denote good performance). From Table 1 and Fig. 4, we note that $\mathrm{L1}_{\mathrm{mask}}$ and $\mathrm{L2}_{\mathrm{mask}}$ perform among the worse, possibly because masks are ill-defined for the silent regions of the mixture. $\mathrm{L1}_{\mathrm{time}}$ and $\mathrm{L2}_{\mathrm{time}}$ also perform poorly, but other time-domain losses like $\mathrm{SISDR}_{\mathrm{time}}$, $\mathrm{SDSDR}_{\mathrm{time}}$, $\mathrm{LOGL1}_{\mathrm{time}}$ and $\mathrm{LOGL2}_{\mathrm{time}}$ perform well. We hypothesise that the logarithmic compression present in those losses helps attenuating low-energy artifacts, like the phase artifacts typically introduced by mask-based models. We also observe that the logarithmic compression might also help training, given that $\mathrm{L1}_{\mathrm{time}}$ and $\mathrm{L2}_{\mathrm{time}}$ metrics are surprisingly bad. This denotes that the model did not train as anticipated, since one expects a model trained on a specific loss to perform reasonably on a metric based on such loss. We also note that $\mathrm{SISDR}_{\mathrm{freq}}$, $\mathrm{LOGL1}_{\mathrm{freq}}$ and $\mathrm{LOGL2}_{\mathrm{freq}}$ perform well. Hence, scale invariant losses and their variants based on LOGL2 and LOGL1 are consistently delivering good results for

---

[2]Rows in Fig. 4 are standardized at zero mean and unit variance and clipped at -1 and 1 for ease of visualization.

	Vocals	Drums	Bass	Other	Mean
L1$_{freq}$	5.95	5.58	4.24	3.90	4.92
L2$_{freq}$	6.40	5.86	5.28	**4.57**	**5.53**
L1$_{mask}$	5.10	4.14	2.88	2.53	**3.66**
L2$_{mask}$	4.74	4.39	2.41	2.71	**3.56**
L1$_{time}$	4.63	4.80	3.53	3.01	**3.99**
L2$_{time}$	4.55	4.25	3.06	2.92	**3.70**
SISDR$_{time}$	6.24	5.76	5.06	4.37	5.35
SISDR$_{freq}$	6.26	6.09	**5.55**	4.54	**5.61**
SDSDR$_{time}$	6.02	5.84	4.99	4.37	5.30
LOGL1$_{time}$	5.95	5.82	5.23	4.35	5.34
LOGL2$_{time}$	5.88	5.81	5.04	4.10	5.21
LOGL1$_{freq}$	6.28	5.90	5.49	4.44	**5.53**
LOGL2$_{freq}$	6.15	5.79	5.38	4.36	5.42
PSA	6.18	**6.17**	5.10	4.33	5.44
Dissim$_{freq}$	6.04	5.66	5.17	4.38	5.31
MRS	5.82	5.11	4.48	3.57	4.80
DeepFeature	6.14	5.89	4.80	4.30	5.28
Adversarial	**6.50**	6.15	5.20	4.47	**5.58**
Combination	5.38	5.43	3.94	3.22	4.49

Table 1. Signal-to-distortion (SDR, in dB; the higher the better).
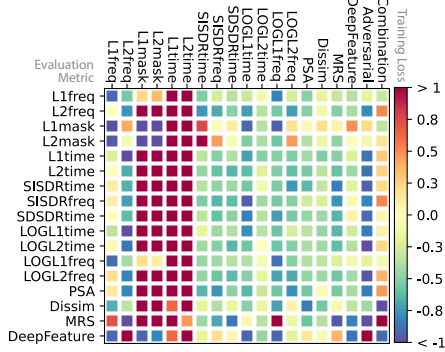
**Fig. 4**. Loss-based metrics[2] over the test set (the lower the better).

both time-domain and spectrogram signals. The widely used L2$_{freq}$ loss also achieves good results, also when coupled with adversarial training and with the phase-sensitive loss (PSA). Yet, its separations did not improve when combined with deep feature losses. We studied combining the most promising losses (Combination: L2$_{freq}$+SISDR$_{freq}$+LOGL1$_{freq}$) with our regularizers (dissimilarity, adversarial and deep feature losses) without much success.

5. SUBJECTIVE EVALUATION

Since running a subjective evaluation for all losses is barely practicable, we relied on the results above and informal listening to select the five best performing ones: LOGL1$_{freq}$, LOGL1$_{time}$, Adversarial, L2$_{freq}$ and SISDR$_{freq}$. For each loss, we evaluated vocals, bass, drums and 'other' separations for 4 different songs. 15 participants were asked to rate the global quality of the separations using a 0–100 scale [36]. The mean opinion scores (MOS) are reported in

	L2$_{freq}$	SISDR$_{freq}$	LOGL1$_{time}$	LOGL1$_{freq}$	Adv
Vocals	57.24$_{\pm 17.9}$	53.33$_{\pm 17.0}$	54.83$_{\pm 19.1}$	**59.10**$_{\pm 17.0}$	54.42$_{\pm 18.4}$
Drums	**60.58**$_{\pm 15.6}$	59.60$_{\pm 18.5}$	59.10$_{\pm 19.8}$	59.76$_{\pm 21.7}$	54.40$_{\pm 18.2}$
Bass	61.62$_{\pm 20.3}$	62.80$_{\pm 19.9}$	57.19$_{\pm 23.2}$	62.13$_{\pm 22.5}$	**64.28**$_{\pm 21.0}$
Other	**56.90**$_{\pm 18.6}$	52.22$_{\pm 19.0}$	48.98$_{\pm 18.5}$	54.88$_{\pm 18.5}$	48.37$_{\pm 20.4}$
Mean	**59.09**$_{\pm 18.2}$	56.99$_{\pm 19.0}$	55.01$_{\pm 20.5}$	58.96$_{\pm 20.1}$	55.39$_{\pm 20.2}$

Table 2. MOS ($\pm$ standard deviation) subjective test.
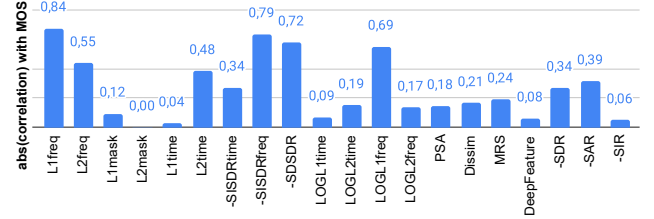
Fig. 5. Correlation between the loss-based metrics (Fig. 4, rows) and the MOS human ratings (Table 2, rows).

Table 2. We find that LOGL1$_{freq}$ and L2$_{freq}$ perform similarly (t-test: $p = 0.94$) and outperform the rest (t-test: $p < 0.05$ except SISDR$_{freq}$ (t-test: $p < 0.27$). Yet, overall, all separations lie in the upper *fair* (40–60) range. Notably, recent music source separation studies also reported MOS scores around the *fair* (40–60) and *good* (60–80) range [10]. Hence, despite recent progress, the task is far from being solved. Importantly, if we compare the results of this subjective test with the evaluation metrics in section 4, we see that it exists a ≈ 0.1 dB SDR difference between L2$_{freq}$ (best human rating) and SISDR$_{freq}$ (best SDR score). This discrepancy is remarkable, since a ≈ 0.1 dB improvement is often enough to claim state-of-the-art results in the literature [10, 37, 38]—although L2$_{freq}$ (with lower SDR) seems to be perceptually preferable. Hence, does it exist a metric that correlates better with human judgment than SDR? To answer this question, we cross-correlate our loss-based metrics (Fig. 4, rows) with the results of the subjective test (Table 2, rows). This experiment (Fig. 5) shows that spectrogram-based metrics, like L1$_{freq}$ or SISDR$_{freq}$, are the ones correlating best with human judgment.

6. DISCUSSION

We extensively reviewed the most representative audio source separation losses and benchmarked those for the task of music source separation. After evaluating those objectively and subjectively, we recommend training with the following spectrogram-based losses: L2$_{freq}$, SISDR$_{freq}$, LOGL2$_{freq}$ or LOGL1$_{freq}$ with, potentially, phase-sensitive objectives and adversarial regularizers. We also found that LOGL1$_{time}$ can deliver competent results, and we do not recommend using mask-based losses. That said, we want to emphasize the limitations of our experimental setup. Since our experiments rely on OpenUnmix, a spectrogram-based model predicting mask filters, our conclusions might be constrained to this family of models. For example, while L1$_{time}$ and L2$_{time}$ performed poorly in our setup, these are widely used to train waveform-based models [10, 11, 39]. Finally, we also cross-correlated the set of metrics we investigated with the results of our subjective test. Out of this experiment, we conclude that it could be informative if future works on music source separation also reported spectral distortion metrics (like L1$_{freq}$) together with SDR-based ones. Note that such metrics are already being used to evaluate speech synthesis models [40].

7. REFERENCES

[1] J. Le Roux, S. Wisdom, H. Erdogan, and J. R. Hershey, "SDR – half-baked or well done?," *ICASSP*, 2019.

[2] B. Kadıoğlu, M. Horgan, X. Liu, J. Pons, D. Darcy, and V. Kumar, "An empirical study of conv-tasnet," *ICASSP*, 2020.

[3] X. Liu and J. Pons, "On permutation invariant training for speech source separation," *ICASSP*, 2021.

[4] E. Cano, D. FitzGerald, and K. Brandenburg, "Evaluation of quality of sound source separation algorithms: Human perception vs quantitative metrics," in *EUSIPCO*, 2016.

[5] 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.

[6] P. Chandna, M. Miron, J. Janer, and E. Gómez, "Monoaural audio source separation using deep convolutional neural networks," *International Conference LVA/ICA*, 2017.

[7] A. Roebel, J. Pons, M. Liuni, and M. Lagrangey, "On automatic drum transcription using non-negative matrix deconvolution and itakura saito divergence," *ICASSP*, 2015.

[8] C. Févotte, N. Bertin, and J. Durrieu, "Nonnegative matrix factorization with the itakura-saito divergence: With application to music analysis," *Neural computation*, 2009.

[9] H. Zen and A. Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," *ICASSP*, 2014.

[10] A. Défossez, N. Usunier, L. Bottou, and F. Bach, "Music source separation in the waveform domain," *arXiv*, 2019.

[11] J. Pons, S. Pascual, G. Cengarle, and J. Serrà, "Upsampling artifacts in neural audio synthesis," *ICASSP*, 2021.

[12] E. Vincent, R. Gribonval, and C. Févotte, "Performance measurement in blind audio source separation," *IEEE/ACM TASLP*, 2006.

[13] Y. Luo and N. Mesgarani, "Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation," *IEEE/ACM TASLP*, 2019.

[14] J. Heitkaemper, D. Jakobeit, C. Boeddeker, L. Drude, and R. Haeb-Umbach, "Demystifying tasnet: A dissecting approach," *arXiv*, 2019.

[15] H. Erdogan, J. Hershey, S. Watanabe, and J. Le Roux, "Phase-sensitive and recognition-boosted speech separation using deep recurrent neural networks," *ICASSP*, 2015.

[16] E. Gusó, "On loss functions for music source separation," *Master Thesis, Universitat Pompeu Fabra*, 2020.

[17] P. Huang, M. Kim, M. Hasegawa-Johnson, and P. Smaragdis, "Deep learning for monaural speech separation," *ICASSP*, 2014.

[18] R. Yamamoto, E. Song, and J. Kim, "Parallel wavegan: A fast waveform generation model based on generative adversarial networks with multi-resolution spectrogram," *ICASSP*, 2020.

[19] S. Arık, H. Jun, and G. Diamos, "Fast spectrogram inversion using multi-head convolutional neural networks," *IEEE SPL*, 2018.

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, 2020.

[21] S. Pascual, A. Bonafonte, and J. Serrà, "SEGAN: Speech enhancement generative adversarial network," *Interspeech*, 2017.

[22] D. Stoller, S. Ewert, and S. Dixon, "Adversarial semi-supervised audio source separation applied to singing voice extraction," *ICASSP*, 2018.

[23] X. Mao, Q. Li, H. Xie, R. Lau, Z Wang, and S. Smolley, "Least squares generative adversarial networks," *ICCV*, 2017.

[24] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," *NeurIPS*, 2016.

[25] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," *CVPR*, 2016.

[26] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv*, 2015.

[27] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," *European Conference on Computer Vision*, 2016.

[28] A. Sahai, R. Weber, and B. McWilliams, "Spectrogram feature losses for music source separation," *EUSIPCO*, 2019.

[29] S. Hershey, S. Chaudhuri, D. Ellis, J. Gemmeke, R. Jansen, A .and Moore, M. Plakal, D. Platt, R. Saurous, and B. Seybold, "Cnn architectures for large-scale audio classification," *ICASSP*, 2017.

[30] J. Gemmeke, D. Ellis, D. Freedman, A. Jansen, W. Lawrence, R. Moore, M. Plakal, and M. Ritter, "Audio set: An ontology and human-labeled dataset for audio events," *ICASSP*, 2017.

[31] Z. Rafii, A. Liutkus, F. Stöter, S. Mimilakis, and R. Bittner, "The MUSDB18 corpus for music separation," Dec. 2017.

[32] T. Bertin-Mahieux, D.P.W. Ellis, B. Whitman, and P. Lamere, "The million song dataset," *ISMIR*, 2011.

[33] T. Chan, T. Yeh, Z. Fan, H. Chen, L. Su, Y. Yang, and R. Jang, "Vocal activity informed singing voice separation with the ikala dataset," *ICASSP*, 2015.

[34] E. Fonseca, J. Pons, X. Favory, F. Font, D. Bogdanov, A. Ferraro, S. Oramas, A. Porter, and X. Serra, "Freesound datasets: a platform for the creation of open audio datasets," *ISMIR*, 2017.

[35] D. Ward, R. D Mason, C. Kim, F. Stöter, A. Liutkus, and M. Plumbley, "Sisec 2018: State of the art in musical audio source separation-subjective selection of the best algorithm," *WIMP*, 2018.

[36] M. Schoeffler, S. Bartoschek, F. Stöter, M. Roess, S. Westphal, B. Edler, and J. Herre, "webmushra—a comprehensive framework for web-based listening tests," *Journal of Open Research Software*, vol. 6, no. 1, 2018.

[37] T. Li, J. Chen, H. Hou, and M. Li, "Sams-net: A sliced attention-based neural network for music source separation," *ISCSLP*, 2021.

[38] N. Takahashi and Y. Mitsufuji, "D3net: Densely connected multidilated densenet for music source separation," *arXiv*, 2020.

[39] D. Stoller, S. Ewert, and S. Dixon, "Wave-u-net: A multi-scale neural network for end-to-end audio source separation," *arXiv*, 2018.

[40] R. Kubichek, "Mel-cepstral distance measure for objective speech quality assessment," *PACRIM*, 1993.