基于深度学习的音轨分离研究报告

何雨洲

概述

课题：通过分析音乐音频的频域或者时域特征进行深度学习，输出分离后的音轨。研究主要内容包括前期调研，模型结构、损失函数的选择，训练方法的设计和额外数据集的构建等。报告参考了进几年的音轨分离和语音增强领域的论文和开源项目，并概括了一些提高音轨分离表现的方法。

1 音轨分离任务

1.1 方法

声道相减. 声道相减是最简单的音轨分离方法。在人声是单声道，伴奏是立体声的假设下，可以通过提取两声道中相似的部分来消除或者提取人声。在实际应用中此假设经常不成立，比如鼓点通常会双声道一致，某些风格的音乐会含有双声道不同的人声部分等。

传统方法. 常见的解决方案有独立分量分析，稀疏主成分分析，非负矩阵分解等和基于统计模型或者机器学习的最优估计。此实验主要分析深度学习方法，不做传统方法的展开。

基于深度学习. 相比其他的方法，深度学习能够做到更精确的分离，在大量的数据集下能做到较强的泛化能力。基于深度学习的音轨分离也是此报告的重点。

1.2 对比语音分离任务

语音分离任务通常分为三类：语音增强、多说话人分离和解混响。与音轨分离一样，深度学习在语音分离任务中表现优秀。音轨分离和语音增强任务有一定的相似度。比如 demucs 网络原用于音轨分离，在只减少卷积层参数不改变网络整体结构的情况下接近了语音增强的 SOTA 标准 [4]。同样的，用于语音增强的模型 Conv-TasNet 在同比例增加了采样率和卷积核大小的情况下也在音轨分离中获得了较好的效果。实验也尝试了采用同样的方法将用于语音分离的 DCCRN 模型同比增加采样率和卷积核大小，同样达到了较好的效果。

1.3 对比语义分割任务

语义分割与基于频谱图的音轨分离任务相似。从二维图片中分离物体轮廓可以类比从二维频谱图中分离特定音源的频谱，主要区别在于音轨分离任务中音频的频谱会重叠。语义分割的掩膜输出 0 和 1，而频域音轨分离的掩膜输出经过 sigmoid/softmax 激活后的权重。语义分割模型无法和语音增强模型一样直接用于音轨分离任务，但是语义分割领域的特定模型结构和优化思想对音轨分离有着较大帮助。

1.4 前期调研

商用的音轨分离模型主要基于 MIT 协议开源的 Spleeter 框架，如 Djay Pro AI 的 neuralmix，团子（Dango），Moises。相比于其他更复杂的模型，Spleeter[5]参数量较少且训练、运行速度快，更适合云端调用和本地实时运行。同时 spleeter 拥有较大的开源社区，有基于各种不同平台和语言的部署尝

试。比较开源的 Spleeter 预训练模型，此前例举的商用模型拥有一定程度上更好的表现。团子对于鼓点音色有着更好的还原，neuralmix 则能在某些情况下（比如电音）更准确的分离人声和伴奏。在摇滚或者电子乐等伴奏声音音量大且风格多变的情况下，各种模型的分离效果均带有一定的杂音。虽然带杂音的音轨无法直接用于伴奏，但是对音乐人的扒谱和再创作效率有较大提升。这部分人群同时也是以上商用音轨分离服务的主要用户群体。和 Spleeter 的开源模型一样，大部分音轨分离服务只支持分离人声，伴奏，贝斯，鼓点，钢琴，和其他等 6 类音频。

2 研究准备

2.1 构建数据集

实验主要采用了 musdb18 数据集进行初步实验和复现。Musdb18 是音轨分离领域最大的公开数据集，包含了 150 首歌曲的 stem 文件，训练集总时长约为 10 小时。通过 stem 文件的分离，可生成 150 个包含 “vocal”，“drums”，“bass”，“other”，“mixture”，“linear mixture”，“accompaniment” 单独 wav 文件的歌曲文件夹。通常 100 首用于训练,50 首用于测试。在训练中，输入 mixture 或者 linear mixture 文件，并用分离的音轨文件进行 loss 计算和模型更新。每首歌均为双声道，采样率为 44.1Khz，长度从 18s 到数分钟不等。大部分歌曲为流行和摇滚音乐，语音为英语。

大部分音源分离研究论文的额外训练采用了 Bean 数据集（80 小时左右），但此数据集为私人数据集，并未公开。另外还有例如 Slakh 等大量的合成音乐数据集（不包含人声）。先前实验表明合成音乐在乐器分离任务中有一定的作用。相比只使用 5 小时的 musdb18 数据集，额外使用 50 小时的 Slakh 训练使基于 4 层 BLSTM 的模型在 musdb18 测试集上额外提升了 1 左右的 SI-SDR[8]。值得注意的是，比起增加的训练时长，合成音乐对模型的提升非常有限。Slakh 数据集更适合包含合成音乐的分离任务。因此，在训练资源的约束下，此实验没有采用合成音乐数据集。

此实验额外从音乐资源网站下载了总计 80 小时的人声（阿卡贝拉），键盘，鼓点，和贝斯音轨，总时间和 Bean 数据集类似。其中各音轨的比例约为【2：2：3：3】。other 部分使用键盘乐器表示。每条音轨长度在 20s 到数分钟不等，大部分为英语，人声部分主要为流行音乐和摇滚。每个音频文件均为双声道，采样率为 44.1Khz。这种方法的优点在于数据集的获取简单。比起一首歌曲完整的音轨分离文件，单独的音轨文件可以从各大音乐资源网站下载。同时，这种训练方法也可以在完整数据集缺失的情况下支持更多种不同的乐器的分离。

2.2 特征提取

音轨分离可以在时域或者频域进行。为了加速训练，实验将音频输入转换成了单声道，并且降低采样率至 3/4（舍弃频率在 16537Hz 以上的部分）。人耳对于高频声音不敏感，且大部分人很难识别 15000Hz 以上的音频。超过 6000Hz 的部分主要构成音乐中的空气感，明亮度和透明度，对听感影响较小 [11]。因此，也可以降低采样率至 22050Hz（舍弃频率在 11025Hz 以上的部分）来进一步增加训练效率。

频域方法. 利用短时傅里叶变换（STFT）将音频从一维的矩阵变换成二维的频谱，再将混合音轨的频谱和分离音轨的频谱用作训练素材。分离后的音轨频谱可以用短时傅里叶逆变换（iSTFT）重新转变为时域音频信号用于输出。值得注意的是大部分频域方法只保留了频谱图的绝对值或者能量并舍去了相位信息。下面的复数频域方法会介绍相位信息的特征提取。

时域方法. 直接对语音进行端到端的分离。时域方法主要包括两类: direct regression 和 adaptive front-end approaches。其主要区别在于第二种方法使用了类似 STFT 的编码器来提取特征输入下一层分离网络，如 Conv-TasNet，而第一种直接用一维卷积来训练端到端的回归方程，如 wave-u-net[6]。通过

编码器计算特征解决了频域方法因为只使用 STFT 振幅而导致相位信息丢失的问题。值得一提的是，通过初始化成特定参数，第二类方法的编码器能用卷积层模仿 STFT 输出频谱图。利用编码器实现 STFT 的优势在于可以通过深度学习逐步优化生成的频谱图，从而得到更多的有用信息。

复数频域方法. 在一些初期的音源分离论文的影响下，很多研究者认为相位信息是不重要的。[14]。David 在 1985 年的实验中指出对于更精确的相位估计没有提高传统算法的分离结果。随着深度学习在音源分离领域的大量应用，相位的重要性再次被提起。Hyeong-Seok Choi 指出随着信号信噪比的降低，相位对分离后信号的影响会逐渐增加 [1]。Donald 在实验中发现，信噪比在-3db 和 0db 的情况下各类相位提取方法都可以提升分离的 SNR。但是，在信噪比 3db 的低噪音环境下，相位对分离后 SNR 的提升作用较小。[16] 作者也提到，相位的优化对于 SNR 和 PESQ 等指标的提升较小（尤其是 SNR），但是在听感的提升上较大。

在使用 Spleeter 进行实验后，发现使用理想相位和混合相位的分离结果在听感上几乎没有差别。在尝试语音分离任务后发现，比起音源分离，相位的作用在高噪音或者多说话人的语音增强任务中更明显。目前相位的重要性均由消融实验体现，缺少严谨的理论证明。有实验提出相位的重要性可能与相位谱的群时延有关。[15]

2.3 模型选择

实验测试了 SpleeterNet，Phasen，DCCRN，Con-TasNet，D3Net，Demucs 等开源模型。此部分主要介绍各模型的特点和可以参考的结构。

Spleeter[5]. 主要特点: U-Net。使用了常见的 Unet，Skip-Connection 和卷积结构。Unet 用于提取不用尺度下的特征信息，skip-connection 用于复用特征，卷积实现上下采样。在输入频谱图的振幅（舍弃相位）后，Spleeter net 输出一张尺寸与输入频谱相同的掩膜。在相位信息不重要的假设下，使用分离前的相位乘以分离后的振幅后可以得到分离后的频谱。在官方开源程序中，4 轨及以下的分离任务使用 Sigmoid 作为掩膜的激活函数，而 5 轨（额外增加钢琴）的分离任务使用 Softmax 作为掩膜的激活函数。Sigmoid 将混合音频的分离任务转换成了【包含乐器 A：不包含乐器 A 】的二分类问题，而 Softmax 则将其转换成了【乐器 A：B：C：D】的多分类问题，并严格限制所有的音轨相加的和等于混合音轨。相比 Softmax，Sigmoid 函数应用更广。在不受其他音轨分离结果限制的情况下，模型的拆分，训练和使用可以更自由。在此实验中，所有激活函数均为 Sigmoid。在某些情况下，Softmax 的约束使模型可以更快的收敛。在测试下，开源 Spleeter 模型中默认的 unet 通道数为【16，32，64，128，256，512】。实验也尝试了额外添加一层通道数为 1024 的卷积来处理欠拟合的问题，在自建数据集上效果较好。

Phasen[18]. 主要特点：two stream blocks (TSB) 和 frequency transformation blocks (FTBs)。模型包括 stream A 和 stream P 两条 stream，分别用于预测振幅和相位，stream A 使用 2 层 FTB（attention+全连接层）包夹 3 层 Conv2d 层，分别用于全局信息和局部信息。stream P 只使用 2 层 Conv2d 更加轻量，第二层卷积使用大小为 25 的卷积核用于提取全局信息。Phasen 在 AudioSet+AVSpeech 数据集上的语音分离任务中的表现超过了 conv-tasnet 2.6 SDR。经过一系列消融实验的比对，Phasen 也指出了 TSB 所提取的相位信息和 FTB 得到的全局信息对结果的 SDR 提升帮助很大。

DCCRN[6]. 主要特点：复数频谱加端对端。来自语音增强领域，在 interspeech2020 取得实时组第一的成绩。DCCRN 采用了一维卷积将时域信号编码为二维频谱并初始化成 STFT，再对复数频谱进行分离。其结构与 Spleeter 类似。DCCRN 的作者提出了一种计算 complex idea ratio mask 的方法，将频谱的实数信息和虚数信息拼接输入网络，同时修改了 LSTM 和 CNN 的结构对这种复数频谱的运算进行了支持。这种 complex idea ratio mask 的计算

方法会同时在参考相位和振幅的频谱输出分离结果，在其他论文中也被用到。

Conditioned-U-Net[9]. 主要特点：Feature-wise Linear Modulation（FiLM）。CU-NEt 的主要框架和 Spleeter 类似为 U-net，但是在每个下采样层的 BN 和 Relu 层之间都加了 FiLM 层。特征输入会经过 $FiLM(x) = \gamma(z) \cdot x + \beta(z)$ 变换，而参数 γ 和 β 由 one-hot 后编码的乐器信息训练。每种不同的乐器对应不同的 γ 和 β 值。作者提出了两种 FiLM 的实现方法：1）简单：所有特征通道共有同一组参数，减少了需要训练的参数量 2）复杂：每个特征通道使用不同参数，增加了模型的自由度。在实验中，使用简单 FiLM 实现的 CU-net 的表现最好，作者猜测更少的参数让模型更容易优化。CU-Net 的分离表现和纯 U-Net 类似，但是 FiLM 层的应用将音源分离任务需要的多个子模型简化成了单个模型，减少了最终模型的大小。

LASAFT[2]. 主要特点：Gated Point-wise Convolutional Modulation（GPoCM），Time-Distributed Fully-connected layers (TDF), Latent Source Attentive Frequency Transformation （LASAFT）。TDF 由全连接层，BN 和 Relu 构成，用于提取频谱图上的全局信息。GPoCM 在 FiLM 的基础上将控制部分修改为

$$GPoCM(X_c^i|\omega_c^i, \beta_c^i) = \sigma(\beta_c^i + \sum_j \omega_{cj}^i \cdot X_j^i) \circ X_c^i$$

X 是每个解码器（上采样）的输出，ω 是待训练的参数，σ 是 sigmoid 激活函数，，$\circ$ 是 Hadamard-product。GPoCM 被放在 Unet 输出端每一个解码器之后。相比只参考当层 unet 特征的 FiLM，GPoCM 结合了不同尺度下的特征。作者表示 GPoCM 有更强的表达能力和灵活性。

作者在实验后发现，前文提到的 TDF 并不能很好的概括不同乐器之间类似的部分，于是设计了 LASAFT 层。LASAFT 被放在每个 TDF 和 GPoCM 之后，它使用 attention 机制，从不同乐器的频率相关性和当前乐器的编码来判断如何注意其他的乐器。LASFT 模型达到了较高的 SDR，尤其是 other 部分。作者表示 other 部分乐器种类较多，而 LASAFT 具有从其他乐器中提取信息的注意力机制。

D3Net[12]. 主要特点：Dense Block，频率带，空洞卷积。D3Net 基于同作者的 MMDenseNet 和 MMDenseLSTM。前者提出了频段切割的方法，在不同的频率段使用不同深度的网络训练，再用 dense block 将拼接的结果整合成输出。Dense block 可以重复使用特征，从而增加训练效率，减少参数量，并提高模型表达能力。MMdenseNet 的下采样部分使用了 1x1 卷积核池化层，并在池化层之间使用 dense block 提取特征。后者在 MMdenseNet 的基础上添加了 LSTM。为了避免模型尺寸过大，作者仅在 U-Net 最底层的上采样前添加了 LSTM。相比 MMdenseNet，MMDenseLSTM 在 vocals 和 other 部分 SDR 提升明显。

D3Net 则在 MMdenseNet 的基础上将 Dense Block 中的二维卷积改成了空洞卷积，并命名为 D2 Block。多个 D2 Block 用同样的 Skip Connection 组成了 D3 Block，用来代替 MMdenseNet 中的 Dense Block。由于大量的 Dense Connection，D3Net 的参数量很小，但是计算速度比较缓慢，在 CPU 上的推导速度严重低于实时。D3Net 和 MMdenseLSTM 都在 musdb18 的分类任务上达到了较好的表现，尤其是人声和鼓的分类。

Conv-TasNet[17]. 基于同作者的 TasNet，并将其中的 LSTM 层改进成了 TCN（时域卷积网络）层，从而做到了只使用卷积层提取和分析特征。相比 LSTM，TCN 的显存需求更少速度更快，同时解决了 LSTM 等递归网络的并发问题。TCN 使用了扩张卷积来提取从长时间序列提取特征，从而保证了足够大的感受野。在实验中发现，conv-TasNet 存在缺音的问题，音轨的某些部分断断续续。

Demucs[4]. 主要特点：双向 LSTM。结构类似于 conv-Tasnet。在其基础上，Demucs 使用了双向 LSTM 提取时序信息，在消融实验对比中提升了

0.88 的 SDR（对比额外添加一层编解码器的无 LSTM 模型只有 0.34 SDR 的提升）。在输入端使用 sinc interpolation 提高采样率并在输出端降低采样率后，模型效果提高了 0.25。在实验中发现，demcus 和 conv-Tasnet 一样存在一定的缺音问题。

模型总结.

- 可以使用 dense block 代替 Conv2d 层，也可以使用多个 dense block 来增强模型表达力
- 可以使用空洞卷积/attention/Fully-Connected Layer 提取全局信息
- 可以使用时域/复数频域方法避免丢失相位信息
- 可以使用 Conditioned Unet 缩小模型大小并提取其他音轨中的信息。
- 可以使用 LSTM、TCN、Attention 提取时间序列上的信息（如旋律）

2.3.1 损失函数

频域方法通常用 l1 loss (mae loss)，weighted l1 loss，smooth l1 loss 或者 l2 loss (mse loss) 来计算损失。l1 loss 计算损失的绝对值，l1 weighted loss 在 l1 loss 的基础上乘上了标签频谱的振幅。l2 loss 计算损失的平方。相比 l1 loss，l2 loss 虽然对离群点更加敏感，但是收敛速度快，中心可导，有较稳定的解。smooth l1 loss 则参考了 l1 与 l2 的优缺点，并在【-1：1】的区间内用 l2 loss 代替 l1 loss。实践中发现 Spleeter 在 musdb18 上用 l2 loss 收敛更快。值得一提的是，如果使用自建数据集或者交换音轨的数据增强方法，l2 loss 会导致模型难以收敛，可能是因为数据异常点较多。在自建数据集上，使用 smooth l1 loss 达到了更好的效果。

时域方法通常使用 SI-SDR，SI-SNR 和 l1 loss，mse 等指标来计算损失。由于模型在数据集上的分离效果通常由 SI-SDR 表示，使用 SI-SDR 或者 SI-SNR 作为指标可以得到更好的分数。l1 和 l2 loss 在准确率上没有明显差别。[4]。

实验尝试了在损失计算阶段将频谱图转变成梅尔刻度。梅尔刻度考虑了人耳听觉对不同频率段敏感程度，使不同的频段在 loss 的计算中获得不同的权重。经过试验，这种方法对分离效果没有明显提升。

3 训练方法

3.1 模型分割

音轨拆分. 音轨分离任务中每条音轨都需要单独的模型（使用 CU-Net 等 conditioned 结构的模型除外）。可以一次只训练针对某一音源的分离模型，再将它们打包成一个完整的模型。音轨拆分有以下几个优点：1）可以单独监控不同音源分离任务的训练情况，使用不同的模型结构和超参数，并且酌情增加或减少迭代次数和数据量，从而达到更高的训练效率和精度。2）减少单次训练过程的显存占用。

频域拆分. 在音轨分离任务中，大部分的信息堆积在低频部分，尤其是贝斯，人声，和鼓。提升低频部分的分离效果对总体的分离效果有着较大的帮助。很多结构复杂的模型如 D3Net[12]采用了频域上的拆分，对频率在 2756hz 以下的音频使用了更复杂的模型进行更精准的分离。（将 44.1khz 的音频转换成高度为 2048 的频谱图后取 0：256 的部分，相当于 $22050/8 = 2756$Hz）同样的，也可以像上一部分一样使用不同的模型和超参数来分别训练这些不同弄个的频域带，从而提升模型精度。为了保证全局信息的学习，D3Net 也用较少的参数和分割前的频谱训练了额外的子模型。最后，这些子模型的结果会在拼接后通过额外的层（D3Net 使用了 dense block）整合成输出。

3.2 混合精度训练

为了加速训练过程并减少显存利用，实验中除了 LSTM 层之外的其他层均利用 pytorch 的 autocast 功能从默认的 32 位精度运算降低到了包含 16 位精度的混合运算。通过减少缓存使用并加大 batch size，各种模型在训练中均实现了 80% 以上的加速。直接

使用混合精度对频域模型的精准度影响较小，虽然部分在使用混合精度后出现了数值溢出。在找出发生问题的层并将变量手动转回为 32 位全精度进行规避后，仍能实现混合精度训练。混合精度训练适合前期验证模型，在算力显存足够的情况下全精度训练是最佳选择。

3.3 初始化

常见的初始化方式包括 kaiming initialization 或者 xavier initialization。文中介绍的大部分模型都直接使用这两种初始化方法训练。值得一提的是，demucs 在 kaiming initialization 的基础上以一定比例放大了初始权重的大小，得到了 0.4 SDR 的提升 [4]。

3.4 超参数

由于显存的限制，实验中模型训练的 batch 大小在 2 到 4 之间。为了使训练更加稳定，模型参数在一定的 batch 之后才会更新，从而伪实现更大的 batch 大小。

实验中优化算法均采用了 Adam[7]。在混合精度的训练中，adam 的 eps 参数也从 1e-8 调整为了 1e-4，从而确保模型不会因为梯度过小停止更新。

实验设置了 milestone 参数，在经过一定的 epoch 后会自动减少默认的 learning rate，帮助模型更好的收敛。经过测试，使用 l2 loss 的 Spleeter 模型在每 50 次迭代后减半学习率可以达到较好的学习效果。

实验中的短时傅里叶变换的参数设定为了 4096 的窗长（frame length）和 1024 的滑动距离（frame step）/ 75% 的覆盖率。频谱图的时间分辨率和频率分辨率受窗长影响。随窗长增加，频率分辨率增加且时间分辨率减少。NFFT 点数和窗长相同。这组 STFT 参数被大部分频域论文采用，包括 Spleeter，LASAFT 和 D3Net，方便进行模型间的比较。

研究表明大部分频域特征分析中最优的 STFT 频谱分布通常是更为稀疏的，因此可以利用梯度下降寻找最合适的 STFT 参数 [19]。也可以像前文提到的那样用一维卷积的时域编码器来代替 STFT。

3.5 数据增强

不考虑额外数据的情况下，传统数据增强方法包括 Channel swapping，Stretch，Pitch shifting，Remixing，Inverse Gaussian filtering，Loudness scaling 等。Laure 在 musdb 和 Bean 数据集上用 U-Net 测试了以上几种数据增强方法的效果，发现它们对 SDR 结果的提升非常有限 [10]。虽然 Stretch 在 Bean 上提升了 0.2 的 SDR，但是它却降低了模型在 musdb 上的表现。值得注意的是数据增强对不同模型的效果有一定的差别。在使用了类似方法的情况下（remixing + swap + loudness scaling），BLSTM 在 musdb 上平均提升了 0.35 的 SDR[13]。此外，Demucs 的作者在使用 pitch shifting 方法后得到了 0.48 的提升。数据增强在特定模型上的效果需要通过实验验证。

Alexandre 在 demucs 的训练中还使用了一种 unlabeled data remix 方法 [3]。先用分类器检测一段未标注的音频，如果没有检测到某种乐器，则从已标注的数据中随机截取一段该乐器的音频。在混合后，他们可以用于那种特定乐器的分离训练。额外添加了 2000 首未标注音乐作为训练材料后，Alexandre 发现这种方法在各模型上都有超过 0.2 的提升。这种方法更是使 demucs 在 musdb18 bass 的分离任务上获得了 1.1 SDR 的提升。

3.6 数据集

实验中每个 batch 的训练程序的输入设定为固定值。在频域方法中，输入为 batch 大小 x 声道数（1）x 频谱图大小（512 x 1536）。在时域方法中，输入为声道数（1）x 采样点数量 (通常为 6s x 采样率)。

除了 musdb18, 实验使用了额外的数据集。通过随机混合额外的单轨音频来得到用于训练的混合音频，方法与数据增强的 Remixing 方法类似。与 demucs 所使用的 unlabeled data remix 不同的是，每条音轨都来自不同的歌曲。在初步实验中发现，这种方法制作额外数据集是有效的。使用 20 小时的混

合音频训练后，模型能够较好的分离音轨。对比使用 musdb18 训练的模型，分离后的人声包含了更多的背景音乐残留和杂音，且总体音量较小（约 60%）。

实验发现，只使用额外数据集训练的模型较 musdb18 训练的模型相比 SDR 低了 2。初步推断原因为总音量的降低。尝试将分离后音频的音量整体增加后，SDR 的差没有改变。同时使用额外数据集和 musdb18 训练的模型的 SDR 也低于仅使用 musdb18 训练的模型。

通过 remix 制作的额外数据集的作用有待进一步验证。效果降低的原因可能包括：不同音轨的节奏和整体音量不协调，模型欠拟合。可以尝试其他 remix 方法，并且用更深的模型处理欠拟合问题。

3.7 训练加速

在固定输入和网络结构后，可以通过设置 $torch.backends.cudnn.benchmark = True$（默认 false）来提升网络运行速度。如果不需要复现结果或者基于某一随机数种微调，也可以通过随机 $torch.backends.cudnn.deterministic = False$ 来进一步提高训练速度。

在实验中，部分模型出现了 CPU 上推导缓慢的问题。问题来自模型参数中的 nans 异常数值。使用 $torch.set\_flush\_denormal = True$ 命令之后解决。

参考文献

[1] H.-S. Choi, J.-H. Kim, J. Huh, A. Kim, J.-W. Ha, and K. Lee. Phase-aware speech enhancement with deep complex u-net network, 2019. 3

[2] W. Choi, M. Kim, J. Chung, and S. Jung. Lasaft: Latent source attentive frequency transformation for conditioned source separation, 2021. 4

[3] A. Défossez, N. Usunier, L. Bottou, and F. Bach. Demucs: Deep extractor for music sources with extra unlabeled data remixed, 2019. 6

[4] A. Défossez, N. Usunier, L. Bottou, and F. Bach. Music source separation in the waveform domain, 2019. 1, 4, 5, 6

[5] 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, 5(50):2154, 2020. Deezer Research. 1, 3

[6] Y. Hu, Y. Liu, S. Lv, M. Xing, S. Zhang, Y. Fu, J. Wu, B. Zhang, and L. Xie. Dccrn: Deep complex convolution recurrent network for phase-aware speech enhancement, 2020. 2, 3

[7] D. P. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. arXiv e-prints, Dec 2014. 6

[8] E. Manilow, G. Wichern, P. Seetharaman, and J. L. Roux. Cutting music source separation some slakh: A dataset to study the impact of training data quality and quantity. 2019 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, 2019. 2

[9] G. Meseguer-Brocal. Conditioned-u-net: Introducing a control mechanism in the u-net for multiple source separations, 2019. 4

[10] L. Prétet. Singing voice separation: A study on training data, 2019. 6

[11] W. B. Snow. Audible frequency ranges of music, speech and noise. The Bell System Technical Journal, 10(4):616–627, 1931. 2

[12] N. Takahashi and Y. Mitsufuji. D3net: Densely connected multidilated densenet for music source separation. 2021. 4, 5

[13] S. Uhlich. Improving music source separation based on dnns through data augmentation and network blending, 2017. 6

[14] D. Wang and J. Lim. The unimportance of phase in speech enhancement. IEEE Transactions on Acoustics, Speech, and Signal Processing, 30(4):679–681, 1982. 3

[15] Y. Wang. 「人耳对相位不敏感」到底是什么意思. 3

[16] D. S. Williamson, S. Member, Y. Wang, and D. Wang. Complex ratio masking for monaural speech separation, 2016. 3

[17] L. Yi and M. Nima. Conv-tasnet: Surpassing ideal time–frequency magnitude masking for speech separation. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 27(8):1256–1266, 2019. 4

[18] D. Yin, C. Luo, Z. Xiong, and W. Zeng. Phasen: A phase-and-harmonics-aware speech enhancement network, 2019. 3

[19] A. Zhao, K. Subramani, and P. Smaragdis. Optimizing short-time fourier transform parameters via gradient descent, 2021. 6